EDM16

June 29 – July 2, 2016

Raleigh

North Carolina, USA

Proceedings of the 9th International Conference on Educational Data Mining

T. Barnes, M. Chi, and M. Feng (Eds)

SAS MARi Blackboard CIVITAS LEARNING

CENGAGE Learning NC STATE UNIVERSITY

International Conference on Educational Data Mining (EDM) 2016
Proceedings of the 9th International Conference on Educational Data Mining
Tiffany Barnes, Min Chi, Mingyu Feng (eds)
Raleigh, June 29-July 2, 2016

Preface

The 9th International Conference on Educational Data Mining (EDM 2016) is held under the auspices of the International Educational Data Mining Society at the Sheraton Raleigh Hotel, in downtown Raleigh, North Carolina, in the USA. The conference, held June 29 - July 2, 2016, follows the eight previous editions (Madrid 2015, London 2014, Memphis 2013, Chania 2012, Eindhoven 2011, Pittsburgh 2010, Cordoba 2009 and Montreal 2008).

The EDM conference is the leading international forum for high-quality research that leverages educational data, learning analytics, and machine learning to answer research questions that shed light on the learning processes. Educational data may come from traces that students leave when they interact with learning management systems, interactive learning environments, intelligent tutoring systems, educational games or when they participate in other data-rich learning contexts. The types of data range from raw log files to data captured by eye-tracking devices or other kind of sensors. The methods used by EDM researchers include analytics, data science, data mining, machine learning, as well as social network analysis, graph mining, recommender systems, and model building.

This years conference features three invited talks by: Rakesh Agrawal, President and Founder of Data Insights Laboratories; Marcia C. Linn, Professor of the University of California at Berkeley; and Judy Kay, Professor of the University of Sydney. Judy Kay's invited paper entitled "Enabling people to harness and control EDM for lifelong, life-wide learning" is also presented in the proceedings. Together with the Journal of Educational Data Mining (JEDM), the EDM 2016 conference supports a JEDM Track that provides researchers a venue to deliver more substantial mature work than is possible in a conference proceedings and to present their work to a live audience. The papers submitted to this track followed the JEDM peer review process; three papers have been accepted to the track and will be presented at the conference. The abstracts of the invited talks and accepted JEDM Track papers can be found in the proceedings. The main conference invited contributions to the Research Track and Industry Track. We received 161 submissions (109 full, 45 short, and 7 industry). We accepted 16 exemplary full papers (15% acceptance rate), 14 full papers (27.5% acceptance rate) and 51 short papers for oral presentation (52% acceptance rate) and an additional 40 for poster presentation. Of the industry papers, 3 were selected for oral presentations and 2 for posters.

This year's best paper and best student paper awards were generously sponsored by the Prof. Ram Kumar Memorial Foundation. The best paper was awarded to the paper entitled "How Deep is Knowledge Tracing?" while the best student paper was awarded to the paper entitled "Calibrated Self-Assessment."

The EDM conference traditionally provides opportunities for young researchers, and particularly for PhD students, to present their research ideas and receive feedback from the peers and more senior researchers. This year, the Doctoral Consortium features 6 such presentations. In addition to the main program, the conference also includes 3 workshops: WS1: Computer-Supported Peer Review in Education (CSPRED-2016), WS2: Writing Analytics, Data Mining, and Writing Studies; WS3: Educational Data Analysis using LearnSphere, and 2 tutorials: T1: SAS Tools for Educational Data Mining, and T2: Massively Scalable EDM with Spark. This year we expand our electronic presence with Whova a social app for conference attendees that provided services for personal scheduling, social linking and personalized recommendations of papers.

We thank the sponsors of EDM 2016 for their generous support: Civitas Learning, Blackboard, MARi, SAS, Cengage Learning and the Prof. Ram Kumar Memorial Foundation. We thank North Carolina State University for their in-kind support, and the Sheraton Downtown Raleigh for their excellent conference services. We also thank the program committee members and reviewers, who with their enthusiastic contributions gave us invaluable support in putting this conference together. Last but not least we thank the organizing team: David Lindrum – Sponsorship Chair; Paul Inventado – Proceedings Chair & Webmaster; Collin Lynch – Posters Chair; Ed Gehringer – Student Volunteer Chair; Sidney D'Mello – DC Chair; Erica Snow and Jonathan Rowe – Workshop & Tutorials Chairs; and Piotr Mitros – Industry Track Chair.

Min Chi	Mingyu Feng	Tiffany Barnes
North Carolina State University	SRI	North Carolina State University
Program Co-chair	Program Co-chair	General Chair

Organization

Conference Chair

Tiffany Barnes	North Carolina State University, USA

Program Chairs

Min Chi	North Carolina State University, USA
Mingyu Feng	SRI International

Workshop and Tutorials Chairs

Jonathan Rowe	North Carolina State University, USA
Erica Snow	SRI

Poster Chair and Local Arrangements Chair

Collin Lynch	North Carolina State University, USA

Doctoral Consortium Chair

Sidney D'Mello	University of Notre Dame, USA

Student Volunteers Chair

Ed Gehringer	North Carolina State University, USA

Industry Track Chair

Piotr Mitros	edX

Sponsors Chair

David Lindrum	Soomo Learning

Proceedings Chair and Webmaster

Paul Salvador Inventado	Carnegie Mellon University, USA

Steering Committee / IEDMS Board of Directors

Ryan Baker	Teachers College, Columbia University, USA
Tiffany Barnes	University of North Carolina at Charlotte, USA
Michel Desmarais	Ecole Polytechnique de Montreal, Canada
Sidney D'Mello	University of Notre Dame, USA
Neil Heffernan III	Worcester Polytechnic Institute, USA
Agathe Merceron	Beuth University of Applied Sciences, Germany
Mykola Pechenizkiy	Eindhoven University of Technology, The Netherlands
John Stamper	Carnegie Mellon University, USA
Kalina Yacef	The University of Sydney, Australia

Program Committee

Lalitha Agnihotri	McGraw Hill Education
Esma Aimeur	University of Montreal
Vincent Aleven	Human-Computer Interaction Institute, Carnegie Mellon University
Ivon Arroyo	Worcester Polytechnic Institute
Mirjam Augstein	Upper Austria University of Applied Sciences, Communication and Knowledge Media
Roger Azevedo	North Carolina State University
Costin Badica	University of Craiova, Software Engineering Department, Romania
Ryan Baker	Teachers College, Columbia University
Tiffany Barnes	North Carolina State University
Joseph Beck	Worcester Polytechnic Institute
Yoav Bergner	Educational Testing Service
Gautam Biswas	Vanderbilt University
Mary Jean Blink	TutorGen, Inc.
Jesus G. Boticario	UNED
François Bouchet	LIP6 - Universit Pierre et Marie Curie
Alex Bowers	Teachers College, Columbia University
Jared Boyce	SRI International
Kristy Elizabeth Boyer	University of Florida
Javier Bravo Agapito	Universidad Autonoma de Madrid
Keith Brawner	United States Army Research Laboratory
Emma Brunskill	Carnegie Mellon University
Renza Campagni	Universit degli Studi di Firenze
Alberto Cano	Department of Computer Sciences and Numerical Analysis
Min Chi	North Carolina State University
Mihaela Cocea	School of Computing, University of Portsmouth
Scott Crossley	Georgia State University
Cynthia D'Angelo	SRI International
Sidney D'Mello	University of Notre Dame
Michel Desmarais	Ecole Polytechnique de Montreal
Hendrik Drachsler	Open University of the Netherlands
Michael Eagle	North Carolina State University
Stephen Fancsali	Carnegie Learning, Inc.
Mingyu Feng	SRI International
Vladimir Fomichov	Faculty of Business Informatics,National Research University Higher School of Economics

Davide Fossati	Carnegie Mellon University in Qatar
April Galyardt	University of Georgia
Carlos Garca-Martnez	Computing and Numerical Analysis Dept. Univ. of Crdoba
Dragan Gasevic	University of Edinburgh
Eva Gibaja	Departmen of Computer Science and Numerical Analysis
Daniela Godoy	ISISTAN Research Institute
Ilya Goldin	Pearson
Yue Gong	CS department, Worcester Polytechnic Institute
José González-Brenes	Pearson
Art Graesser	University of Memphis
Joseph Grafsgaard	North Carolina State University, Computer Science
Philip Guo	University of Rochester
Neil Heffernan	Worcester Polytechnic Institute
Arnon Hershkovitz	Tel Aviv University
Andrew Hicks	North Carolina State University
Roland Hubscher	Bentley University
Vladimir Ivančević	University of Novi Sad, Faculty of Technical Sciences
Mike Joy	University of Warwick
Kenneth Koedinger	Carnegie Mellon University
Irena Koprinska	The University of Sydney
Sotiris Kotsiantis	Univerisity of Patras
Andrew Krumm	SRI International
James Lester	North Carolina State University
Innar Liiv	Tallinn University of Technology
Ran Liu	Carnegie Mellon University
Wei Liu	University of Technology, Sydney
Vanda Luengo	Laboratoire d'informatique de Paris, LIP6, Universit Pierre et Marie Curie
Ivan Luković	University of Novi Sad, Faculty of Technical Sciences
J. M. Luna	Dept. of Computer Science and Numerical Analysis
Mihai Lupu	Vienna University of Technology
Maria Luque	University of Cordoba
Collin Lynch	North Carolina State University
Lina Markauskaite	The University of Sydney
Noboru Matsuda	Carnegie Mellon University
Manolis Mavrikis	London Knowledge Lab
Oleksiy Mazhelis	University of Jyvskyl
Gordon McCalla	University of Saskatchewan
Victor Menendez	Universidad Autnoma de Yucatn
Agathe Merceron	Beuth University of Applied Sciences Berlin
Donatella Merlini	Universit di Firenze
Cristian Mihaescu	University of Craiova
Carlos Monroy	Rice University
Behrooz Mostafavi	North Carolina State University
Bradford Mott	North Carolina State University
Tristan Nixon	University of Memphis
Roger Nkambou	Universit du Qubec Montral (UQAM)
Benjamin Nye	University of Southern California (Institute for Creative Technologies)
Andrew Olney	University of Memphis
Luc Paquette	Teachers College, Columbia University
Abelardo Pardo	The University of Sydney
Zach Pardos	UC Berkeley
Mykola Pechenizkiy	Department of Computer Science, Eindhoven University of Technology

Radek Pelánek	Masaryk University Brno
Niels Pinkwart	Humboldt-Universitt zu Berlin
Paul Stefan Popescu	Faculty of Automation Conputers an Electronics Craiova
Kaska Porayska-Pomsta	London Knowledge Lab
Thomas Price	North Carolina State University
David Pritchard	MIT
Martina Rau	University of Wisconsin - Madison, Department of Educational Psychology
Steven Ritter	Carnegie Learning, Inc.
Robby Robson	Eduworks
Ma. Mercedes T. Rodrigo	Department of Information Systems and Computer Science, Ateneo de Manila University
Cristobal Romero	Department of Computer Sciences and Numerical Analysis
Jos Ral Romero	University of Cordoba
Carolyn Rose	Carnegie Mellon University
Jonathan Rowe	North Carolina State University
Vasile Rus	The University of Memphis
Maria Ofelia San Pedro	Teachers College, Columbia University
Olga C. Santos	aDeNu Research Group (UNED)
George Siemens	UT Arlington
Erica Snow	Arizona State University
John Stamper	Carnegie Mellon University
Jun-Ming Su	Department of Information and Learning Technology, National University of Tainan
Ling Tan	Australian Council for Educational Research
Stefan Trausan-Matu	University Politehnica of Bucharest
Sebastián Ventura	Department of Computer Sciences and Numerical Analysis
Lucian Vintan	"Lucian Blaga" University of Sibiu
Alina Von Davier	Educational Testing Service
Feng-Hsu Wang	Ming Chuan University
Yutao Wang	WPI
Stephan Weibelzahl	Private University of Applied Sciences Gttingen
Fridolin Wild	The Open University
Joseph Jay Williams	Harvard University
Marcelo Worsley	Stanford University
Kalina Yacef	The University of Sydney
Kalina Yacef	The University of Sydney
Michael Yudelson	Carnegie Learning, Inc.
Amelia Zafra Gómez	Department of Computer Sciences and Numerical Analysis
Alfredo Zapata González	Universidad Autonoma de Yucatan
Diego Zapata-Rivera	Educational Testing Service
Marta Zorrilla	University of Cantabria

Sponsors


sas


MARi


Blackboard


CIVITAS
LEARNING


CENGAGE
Learning


Prof. Ram Kumar Memorial Foundation

Awards

EDM 2016 thanks the Prof. Ram Kumar Memorial Foundation for generously sponsoring the 2016 best paper and best student paper awards.

Best papers and exemplary paper selection

A total of 16 exemplary papers were selected by the program chairs as those that represent the best work submitted to EDM 2016. Candidates for exemplary papers were selected among those accepted as full papers using the following criteria: 1) the average ratings of all reviewers and 2) at least one reviewer indicated the paper should be considered for best paper. Program Chairs and the General Chair then performed meta-reviews for all of these papers to make the final exemplary paper selections.

Finally, the Best Paper Committee was formed to review the top papers in the conference to select best paper nominations. We used random selection to divide both the 16 exemplary papers and the 10 committee members into two groups. All members in the same Best Paper Sub-committee received the same 8 exemplary papers together with their reviews. Sub-committee members were asked to rank the three best papers from the 8 papers, and to provide a 1-2 sentence justification for each of the top 3 they chose. Based on these rankings, four Best Paper nominees were selected.

Best Paper Committee:

Koedinger, Kenneth	Pavlik Jr., Philip I.	Aleven, Vincent	Baker, Ryan	Galyardt, April
Goldin, Ilya	Heffernan, Neil	Ritter, Steven	Olney, Andrew	Pechenizkiy, Mykola

Award winners

Best paper	How Deep is Knowledge Tracing? *Mohammad Khajah, Robert Lindsey and Michael Mozer*
Best student paper	Calibrated Self-Assessment *Igor Labutov and Christoph Studer*
Best paper nominees	LIVELINET: A Multimodal Deep Recurrent Neural Network to Predict Liveliness in Educational Videos *Arjun Sharma, Arijit Biswas, Ankit Gandhi, Sonal Patil and Om Deshmukh*
	How to Model Implicit Knowledge? Similarity Learning Methods to Assess Perceptions of Visual Representations *Martina Rau, Blake Mason and Robert Nowak*
	Measuring Gameplay Affordances of User-Generated Content in an Educational Game *Andrew Hicks, Zhongxiu Liu and Tiffany Barnes*

Table of Contents

Invited Talks (abstracts)

Data-Driven Education: Some opportunities and Challenges . . . 2
Rakesh Agrawal

WISE Ways to Strengthen Inquiry Science Learning . . . 3
Marcia Linn

Enabling people to harness and control EDM for lifelong, life-wide learning . . . 4
Judy Kay

JEDM Track Journal Papers (abstracts)

Toward Data-Driven Design of Educational Courses: A Feasibility Study . . . 6
Rakesh Agrawal, Behzad Golshan and Evangelos Papalexakis

Next-Term Student Performance Prediction: A Recommender Systems Approach . . . 7
Mack Sweeney, Jaime Lester, Huzefa Rangwala and Aditya Johri

Exploring the Effect of Student Confusion in Massive Open Online Courses . . . 8
Diyi Yang, Robert Kraut, and Carolyn Rosé

Invited Paper

Enabling people to harness and control EDM for lifelong, life-wide learning . . . 10
Judy Kay

Full Papers

{ENTER}ing the Time Series {SPACE}: Uncovering the Writing Process through Keystroke Analyses . . . 22
Laura Allen, Matthew Jacovina, Mihai Dascalu, Rod Roscoe, Kevin Kent, Aaron Likens and Danielle McNamara

Automatic Gaze-Based Detection of Mind Wandering during Film Viewing . . . 30
Caitlin Mills, Robert Bixler, Xinyi Wang and Sidney D'Mello

Riding an emotional roller-coaster: A multimodal study of young child's math problem solving activities . . . 38
Lujie Chen, Xin Li, Zhuyun Xia, Zhanmei Song, Louis-Philippe Morency and Artur Dubrawski

Joint Discovery of Skill Prerequisite Graphs and Student Models . . . 46
Yetian Chen, José González-Brenes and Jin Tian

Gauging MOOC Learners' Adherence to the Designed Learning Path . . . 54
Daniel Davis, Guanliang Chen, Claudia Hauff and Geert-Jan Houben

Dynamics of Peer Grading: An Empirical Study . . . 62
Luca de Alfaro and Michael Shavlovsky

Sequence Matters, But How Exactly? A Method for Evaluating Activity Sequences from Data . . . 70
Shayan Doroudi, Kenneth Holstein, Vincent Aleven and Emma Brunskill

Measuring Gameplay Affordances of User-Generated Content in an Educational Game . . . 78
Andrew Hicks, Zhongxiu Liu, Michael Eagle and Tiffany Barnes

The Eyes Have It: Gaze-based Detection of Mind Wandering during Learning with an Intelligent Tutoring System 86
Stephen Hutt, Caitlin Mills, Shelby White, Patrick J. Donnelly and Sidney K. D'Mello

How Deep is Knowledge Tracing? 94
Mohammad Khajah, Robert Lindsey and Michael Mozer

Temporally Coherent Clustering of Student Data 102
Severin Klingler, Tanja Käser, Barbara Solenthaler and Markus Gross

Web as a textbook: Curating Targeted Learning Paths through the Heterogeneous Learning Resources on the Web 110
Igor Labutov and Hod Lipson

Calibrated Self-Assessment 119
Igor Labutov and Christoph Studer

MOOC Learner Behaviors by Country and Culture; an Exploratory Analysis 127
Zhongxiu Liu, Rebecca Brown, Collin Lynch, Tiffany Barnes, Ryan Baker, Yoav Bergner and Danielle Mcnamara

Effect of student ability and question difficulty on duration 135
Yijun Ma, Lalitha Agnihotri, Ryan Baker and Shirin Mojarad

Modeling the Influence of Format and Depth during Effortful Retrieval Practice 143
Jaclyn K. Maass and Philip I. Pavlik Jr.

The Apprentice Learner architecture: Closing the loop between learning theory and educational data 151
Christopher Maclellan, Erik Harpstead, Rony Patel and Kenneth Koedinger

Mining behaviors of students in autograding submission system logs 159
Jessica McBroom, Bryn Jeffries, Irena Koprinska and Kalina Yacef

Modelling the way: Using action sequence archetypes to differentiate learning pathways from learning outcomes 167
Kelvin H. R. Ng, Kevin Hartman, Kai Liu and Andy W H Khong

A Coupled User Clustering Algorithm for Web-based Learning Systems 175
Ke Niu, Zhendong Niu, Xiangyu Zhao, Can Wang, Kai Kang and Min Ye

Execution Traces as a Powerful Data Representation for Intelligent Tutoring Systems for Programming 183
Benjamin Paaßen, Joris Jensen and Barbara Hammer

Generating Data-driven Hints for Open-ended Programming 191
Thomas Price, Yihuan Dong and Tiffany Barnes

How to Model Implicit Knowledge? Similarity Learning Methods to Assess Perceptions of Visual Representations 199
Martina Rau, Blake Mason and Robert Nowak

Student Usage Predicts Treatment Effect Heterogeneity in the Cognitive Tutor Algebra I Program 207
Adam Sales, Asa Wilks and John Pane

LIVELINET: A Multimodal Deep Recurrent Neural Network to Predict Liveliness in Educational Videos 215
Arjun Sharma, Arijit Biswas, Ankit Gandhi, Sonal Patil and Om Deshmukh

Semantic Features of Math Problems: Relationships to Student Learning and Engagement 223
Stefan Slater, Jaclyn Ocumpaugh, Ryan Baker, Peter Scupelli, Paul Salvador Inventado and Neil Heffernan

An Ensemble Method to Predict Student Performance in an Online Math Learning Environment 231
Martin Stapel, Zhilin Zheng and Niels Pinkwart

Predicting Post-Test Performance from Student Behavior: A High School MOOC Case Study . . . 239
Sabina Tomkins, Arti Ramesh and Lise Getoor

The Affective Impact of Tutor Questions: Predicting Frustration and Engagement . . . 247
Alexandria Vail, Joseph Wiggins, Joseph Grafsgaard, Kristy Boyer, Eric Wiebe and James Lester

Unnatural Feature Engineering: Evolving Augmented Graph Grammars for Argument Diagrams . . . 255
Linting Xue, Collin Lynch and Min Chi

Short Papers

Investigating Swarm Intelligence for Performance Prediction . . . 264
Mohammad Majid Al-Rifaie, Matthew Yee-King and Mark d'Inverno

Predicting Student Progress from Peer-Assessment Data . . . 270
Michael Mogessie Ashenafi, Marco Ronchetti and Giuseppe Riccardi

Topic-wise Classification of MOOC Discussions: A Visual Analytics Approach . . . 276
Thushari Atapattu, Katrina Falkner and Hamid Tarmazdi

Document Segmentation for Labeling with Academic Learning Objectives . . . 282
Divyanshu Bhartiya, Danish Contractor, Sovan Biswas, Bikram Sengupta and Mukesh Mohania

Semi-Automatic Detection of Teacher Questions from Human-Transcripts of Audio in Live Classrooms . . . 288
Nathaniel Blanchard, Patrick Donnelly, Andrew Olney, Borhan Samei, Sean Kelly, Xiaoyi Sun, Brooke Ward, Martin Nystrand and Sidney D'Mello

Modeling Interactions Across Skills: A Method to Construct and Compare Models Predicting the Existence of Skill Relationships . . . 292
Anthony F. Botelho, Seth Adjei and Neil Heffernan

Robust Predictive Models on MOOCs : Transferring Knowledge across Courses . . . 298
Sebastien Boyer and Kalyan Veeramachaneni

A Comparative Analysis of Techniques for Predicting Student Performance . . . 306
Hana Bydžovská

Course Enrollment Recommender System . . . 312
Hana Bydžovská

Data-driven Automated Induction of Prerequisite Structure Graphs . . . 318
Devendra Singh Chaplot, Yiming Yang, Jaime Carbonell and Kenneth R. Koedinger

Exploring Learning Management System Interaction Data: Combining Data-driven and Theory-driven Approaches . . . 324
Hongkyu Choi, Ji Eun Lee, Won-Joon Hong, Kyumin Lee, Mimi Recker and Andy Walker

A Comparison of Automatic Teaching Strategies for Heterogeneous Student Populations . . . 330
Benjamin Clement, Pierre-Yves Oudeyer and Manuel Lopes

Automatic Assessment of Constructed Response Data in a Chemistry Tutor . . . 336
Scott Crossley, Kris Kyle, Jodi Davenport and Danielle McNamara

Choosing versus Receiving Feedback: The Impact of Feedback Valence on Learning in an Assessment Game . . . 341
Maria Cutumisu and Daniel L. Schwartz

Course Content Analysis: An Initiative Step toward Learning Object Recommendation Systems for MOOC Learners . . . 347
Yiling Dai, Yasuhito Asano and Masatoshi Yoshikawa

Student Emotion, Co-occurrence, and Dropout in a MOOC Context 353
John Dillon, Nigel Bosch, Malolan Chetlur, Nirandika Wanigasekara, G. Alex Ambrose, Bikram Sengupta and Sidney D'Mello

Semi-Markov model for simulating MOOC students 358
Louis Faucon, Łukasz Kidziński and Pierre Dillenbourg

Investigating Gender Difference on Homework in Middle School Mathematics 364
Mingyu Feng, Jeremy Roschelle, Craig Mason and Ruchi Bhanot

Investigating Difficult Topics in a Data Structures Course Using Item Response Theory and Logged Data Analysis 370
Eric Fouh, Mohammed F. Farghally, Sally Hamouda, Kyu Han Koh and Clifford A. Shaffer

Acting the Same Differently: A Cross-Course Comparison of User Behavior in MOOCs 376
Ben Gelman, Matt Revelle, Carlotta Domeniconi, Kalyan Veeramachaneni and Aditya Johri

Collaborative Problem Solving Skills versus Collaboration Outcomes: Findings from Statistical Analysis and Data Mining 382
Jiangang Hao, Lei Liu, Alina von Davier, Patrick Kyllonen and Christopher Kitchen

Hint Availability Slows Completion Times in Summer Work 388
Paul Salvador Inventado, Peter Scupelli, Eric Van Inwegen, Korinn Ostrow, Neil Heffernan III, Jaclyn Ocumpaugh, Ryan Baker, Stefan Slater and Mia Almeda

On Competition for Undergraduate Co-op Placements: A Graph Mining Approach 394
Yuheng Jiang and Lukasz Golab

Expediting Support for Social Learning with Behavior Modeling 400
Yohan Jo, Gaurav Singh Tomar, Oliver Ferschke, Carolyn Rose and Dragan Gasevic

On generalizability of MOOC models 406
Łukasz Kidziński, Kshitij Sharma, Mina Shirvani Boroujeni and Pierre Dillenbourg

Closing the Loop with Quantitative Cognitive Task Analysis 412
Kenneth Koedinger and Elizabeth McLaughlin

Does a Peer Recommender Foster Students' Engagement in MOOCs? 418
Hugues Labarthe, François Bouchet, Remi Bachelet and Kalina Yacef

A Contextual Bandits Framework for Personalized Learning Action Selection 424
Andrew Lan and Richard Baraniuk

How Good Is Popularity? Summary Grading in Crowdsourcing 430
Haiying Li, Zhiqiang Cai and Art Graesser

Beyond Log Files: Using Multi-Modal Data Streams Towards Data-Driven KC Model Improvement 436
Ran Liu, Jodi Davenport and John Stamper

Seeking Programming-related Information from Large Scaled Discussion Forums, Help or Harm? 442
Yihan Lu and Sharon Hsiao

Classifying behavior to elucidate elegant problem solving in an educational game 448
Laura Malkiewich, Ryan S. Baker, Valerie Shute, Shimin Kai and Luc Paquette

Predicting Dialogue Acts for Intelligent Virtual Agents with Multimodal Student Interaction Data 454
Wookhee Min, Joseph Wiggins, Lydia Pezzullo, Alexandria Vail, Kristy Elizabeth Boyer, Bradford Mott, Megan Frankosky, Eric Wiebe and James Lester

Exploring the Impact of Data-driven Tutoring Methods on Students' Demonstrative Knowledge in Logic Problem Solving 460
Behrooz Mostafavi and Tiffany Barnes

Properties and Applications of Wrong Answers in Online Educational Systems 466
Radek Pelánek and Jiří Řihák

Using Inverse Planning for Personalized Feedback 472
Anna Rafferty, Rachel Jansen and Thomas Griffiths

Pattern mining uncovers social prompts of conceptual learning with physical and virtual representations 478
Martina Rau

Predicting Performance on MOOC Assessments using Multi-Regression Models 484
Zhiyun Ren, Huzefa Rangwala and Aditya Johri

Validating Game-based Measures of Implicit Science Learning 490
Elizabeth Rowe, Jodi Asbell-Clarke, Michael Eagle, Andrew Hicks, Tiffany Barnes, Rebecca Brown and Teon Edwards

Assessing Student-Generated Design Justifications in Virtual Engineering Internships 496
Vasile Rus, Dipesh Gautam, Zach Swiecki, David Shaffer and Art Graesser

Tensor Factorization for Student Modeling and Performance Prediction in Unstructured Domain 502
Shaghayegh Sahebi, Yu-Ru Lin and Peter Brusilovsky

Aim Low: Correlation-based Feature Selection for Model-based Reinforcement Learning 507
Shitian Shen and Min Chi

Personalization of Learning Paths in Online Communities of Creators 513
Mingxuan Sun and Seungwon Yang

Modeling Visitor Behavior in a Game-Based Engineering Museum Exhibit with Hidden Markov Models 517
Mike Tissenbaum, Matthew Berland and Vishesh Kumar

Learning Curves for Problems with Multiple Knowledge Components 523
Brett van de Sande

A Nonlinear State Space Model for Identifying At-Risk Students in Open Online Courses 527
Feng Wang and Li Chen

Transactivity as a Predictor of Future Collaborative Knowledge Integration in Team-Based Learning in Online Courses 533
Miaomiao Wen, Keith Maki, Xu Wang, Steven Dow, James Herbsleb and Carolyn Rose

Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation 539
Kevin Wilson, Yan Karklin, Bojian Han and Chaitanya Ekanadham

Going Deeper with Deep Knowledge Tracing 545
Xiaolu Xiong, Siyuan Zhao, Eric Vaninwegen and Joseph Beck

Boosted Decision Tree for Q-matrix Refinement 551
Peng Xu and Michel Desmarais

Individualizing Bayesian Knowledge Tracing. Are Skill Parameters More Important Than Student Parameters? 556
Michael Yudelson

Deep Learning + Student Modeling + Clustering: a Recipe for Effective Automatic Short Answer Grading 562
Yuan Zhang, Rajat Shah and Min Chi

Posters and Demo

Redefining "What" in Analyses of Who Does What in MOOCs . . . 569
Alok Baikadi, Carrie Demmans Epp, Yanjin Long and Christian Schunn

Text Classification of Student Self-Explanations in College Physics Questions . . . 571
Sameer Bhatnagar, Michel Desmarais, Nathaniel Lasry and Elizabeth Charles

Automated Feedback on the Quality of Collaborative Processes: An Experience Report . . . 573
Marcela Borge and Carolyn Rosé

Mining Sequences of Gameplay for Embedded Assessment in Collaborative Learning . . . 575
Philip Buffum, Megan Frankosky, Kristy Boyer, Eric Wiebe, Bradford Mott and James Lester

Can Word Probabilities from LDA be Simply Added up to Represent Documents? . . . 577
Zhiqiang Cai, Haiying Li, Xiangen Hu and Art Graesser

Examining the necessity of problem diagrams using MOOC AB experiments . . . 579
Zhongzhou Chen, Neset Demirci and David Pritchard

Identifying relevant user behavior and predicting learning and persistence in an ITS-based afterschool program 581
Scotty Craig, Xudong Huang, Jun Xie, Ying Fang and Xiangen Hu

Extracting Measures of Active Learning and Student Self-Regulated Learning Strategies from MOOC Data . . . 583
Nicholas Diana, Michael Eagle, John Stamper and Kenneth Koedinger

Exploring Social Influence on the Usage of Resources in an Online Learning Community . . . 585
Ogheneovo Dibie, Tamara Sumner, Keith Maull and David Quigley

Time Series Cross Section method for monitoring students' page views of course materials and improving classroom teaching . . . 587
Konomu Dobashi

Predicting STEM Achievement with Learning Management System Data: Prediction Modeling and a Test of an Early Warning System . . . 589
Michelle Dominguez, Matthew Bernacki and Merlin Uesbeck

Comparison of Selection Criteria for Multi-Feature Hierarchical Activity Mining in Open Ended Learning Environments . . . 591
Yi Dong, John S. Kinnebrew and Gautam Biswas

A Data-Driven Framework of Modeling Skill Combinations for Deeper Knowledge Tracing . . . 593
Yun Huang, Julio Guerra and Peter Brusilovsky

Generating Semantic Concept Map for MOOCs . . . 595
Zhuoxuan Jiang, Peng Li, Yan Zhang and Xiaoming Li

How to Judge Learning on Online Learning: Minimum Learning Judgment System . . . 597
Jaechoon Jo and Heuiseok Lim

Guiding Students Towards Frequent High-Utility Paths in an Ill-Defined Domain . . . 599
Igor Jugo, Božidar Kovačić and Vanja Slavuj

Portrait of an Indexer - Computing Pointers Into Instructional Videos . . . 601
Andrew Lamb, Jose Hernandez, Jeffrey Ullman and Andreas Paepcke

Hierarchical Cluster Analysis Heatmaps and Pattern Analysis: An Approach for Visualizing Learning Management System Interaction Data . . . 603
Ji Eun Lee, Mimi Recker, Alex Bowers and Min Yuan

Understanding Engagement in MOOCs . . . 605
Qiujie Li and Rachel Baker

How quickly can wheel spinning be detected? . . . 607
Noboru Matsuda, Sanjay Chandrasekaran and John Stamper

Exploring and Following Students' Strategies When Completing Their Weekly Tasks . . . 609
Jessica McBroom, Bryn Jeffries, Irena Koprinska and Kalina Yacef

Identifying Student Behaviors Early in the Term for Improving Online Course Performance . . . 611
Makoto Mori and Philip Chan

Time Series Analysis of VLE Activity Data . . . 613
Ewa Młynarska, Pádraig Cunningham and Derek Greene

Massively Scalable EDM with Spark . . . 615
Tristan Nixon

Study on Automatic Scoring of Descriptive Type Tests using Text Similarity Calculations . . . 616
Izuru Nogaito, Keiji Yasuda and Hiroaki Kimura

Equity of Learning Opportunities in the Chicago City of Learning Program . . . 618
David Quigley, Ogheneovo Dibie, Arafat Sultan, Katie Van Horne, William R. Penuel, Tamara Sumner, Ugochi Acholonu and Nichole Pinkard

Novel features for capturing cooccurrence behavior in dyadic collaborative problem solving tasks . . . 620
Vikram Ramanarayanan and Saad Khan

Adding eye-tracking AOI data to models of representation skills does not improve prediction accuracy . . . 622
Martina Rau and Zach Pardos

MATHia X: The Next Generation Cognitive Tutor . . . 624
Steven Ritter and Stephen Fancsali

Towards Integrating Human and Automated Tutoring Systems . . . 626
Steven Ritter, Michael Yudelson, Stephen Fancsali and Susan Berman

Toward Revision-Sensitive Feedback in Automated Writing Evaluation . . . 628
Rod Roscoe, Matthew Jacovina, Laura Allen, Adam Johnson and Danielle McNamara

Preliminary Results On Dialogue Act and Subact Classification in Chat-based Online Tutorial Dialogues . . . 630
Vasile Rus, Rajendra Banjade, Nabin Maharjan, Donald Morrison, Steve Ritter and Michael Yudelson

SAS Tools for Educational Data Mining . . . 632
Jennifer Sabourin, Scott Mcquiggan and André De Waal

Applicability of Educational Data Mining in Afghanistan: Opportunities and Challenges . . . 634
Abdul Rahman Sherzad

Browsing-Pattern Mining from e-Book Logs with Non-negative Matrix Factorization . . . 636
Atsushi Shimada, Fumiya Okubo and Hiroaki Ogata

How employment constrains participation in MOOCs? . . . 638
Mina Shirvani Boroujeni, Łukasz Kidziński and Pierre Dillenbourg

Quantifying How Students Use an Online Learning System: A Focus on Transitions and Performance . . . 640
Erica Snow, Andrew Krumm, Timothy Podkul, Mingyu Feng and Alex Bowers

A Platform for Integrating and Analyzing Data to Evaluate the Impacts of Educational Technologies . . . 642
Daniel Stanhope and Karl Rectanus

Educational Technology: What 49 Schools Discovered about Usage when the Data were Uncovered 644
Daniel Stanhope and Karl Rectanus

Learning curves versus problem difficulty: an analysis of the Knowledge Component picture for a given context 646
Brett van de Sande

Validating Automated Triggers and Notifications @ Scale in Blackboard Learn 648
John Whitmer, Aleksander Dietrichson and Bryan O'Haver

Discovering 'Tough Love' Interventions Despite Dropout 650
Joseph Jay Williams, Anthony Botelho, Adam Sales, Neil Heffernan and Charles Lang

Stimulating collaborative activity in online social learning environments with Markov decision processes 652
Matthew Yee-King and Mark D'Inverno

Predicting student grades from online, collaborative social learning metrics using K-NN 654
Matthew Yee-King, Andreu Grimalt-Reynes and Mark D'Inverno

Meta-learning for predicting the best vote aggregation method: Case study in collaborative searching of LOs ... 656
Alfredo Zapata González, Victor Menendez, Cristobal Romero and Manuel E. Prieto Méndez

Soft Clustering of Physics Misconceptions Using a Mixed Membership Model 658
Guoguo Zheng, Seohyun Kim, Yanyan Tan and April Galyardt

Perfect Scores Indicate Good Students !? The Case of One Hundred Percenters in a Math Learning System 660
Zhilin Zheng, Martin Stapel and Niels Pinkwart

Doctoral Consortium

Towards the Understanding of Gestures and Vocalization Coordination in Teaching Context 663
Roghayeh Barmaki and Charles E. Hughes

Towards Modeling Chunks in a Knowledge Tracing Framework for Students' Deep Learning 666
Yun Huang and Peter Brusilovsky

Using Case-Based Reasoning to Automatically Generate High-Quality Feedback for Programming Exercises 669
Angelo Kyrilov

Predicting Off-task Behaviors for Adaptive Vocabulary Learning System 672
Sungjin Nam

Estimation of prerequisite skills model from large scale assessment data using semantic data mining 675
Bruno Penteado

Designing Interactive and Personalized Concept Mapping Learning Environments 678
Shang Wang

Industry Track - Short Papers

Analysing and Refining Pilot Training 682
Bruno Emond, Scott Buffett, Cyril Goutte and Jaff Guo

A Scalable Learning Analytics Platform for Automated Writing Feedback 688
Jacqueline Feild, Nicolas Lewkow, Neil Zimmerman, Mark Riedesel and Alfred Essa

An Automated Test of Motor Skills for Job Selection and Feedback 694
Bhanu Pratap Singh Rawat and Varun Aggarwal

Industry Track - Posters

Studying Assignment Size and Student Performance Using Propensity Score Matching 701
Shirin Mojarad

Toward Automated Support for Teacher-Facilitated Formative Feedback on Student Writing 703
Jennifer Sabourin, Lucy Kosturko, Kristin Hoffmann and Scott Mcquiggan

TutorSpace: Content-centric Platform for Enabling Blended Learning in Developing Countries 705
Kuldeep Yadav, Kundan Shrivastava, Ranjeet Kumar, Saurabh Srivastava and Om Deshmukh

Invited Talks

(abstracts)

Data-Driven Education: Some opportunities and Challenges

Rakesh Agrawal
Data Insights Laboratories
ragrawal@acm.org

ABSTRACT

A program of study can be viewed as a knowledge graph consisting of learning units and relationships between them. Such a knowledge graph provides the core data structure for organizing and navigating learning experiences. We address two issues in this talk. First, given a knowledge graph, how can we use data mining to identify and correct deficiencies in a knowledge graph. Second, how can we use data mining to form study groups with the goal of maximizing overall learning. We conclude by pointing out some open research problems.

WISE Ways to Strengthen Inquiry Science Learning

Marcia C. Linn
University of California, Berkeley, Berkeley, CA, USA
mclinn@berkeley.edu

ABSTRACT

The Web-based Inquiry Science Environment (WISE) logs student and teacher interactions during classroom science inquiry instruction. Over the past 10 years we have used these logs for many purposes including: to analyze patterns of student interactions with dynamic, interactive scientific models and improve instruction; to determine when students revisit prior activities and assess whether the visit was fruitful; and to analyze the coherence of student essays and offer personalized guidance. I will illustrate our findings with some successes and failures as we attempt to: validate what a sequence of logged actions means; measure a key learning construct with logged interactions; and determine how to use scores derived from logged variables to guide student inquiry learning.

Enabling people to harness and control EDM for lifelong, life-wide learning

Judy Kay
Faculty of Engineering and Information Technologies, The University of Sydney, Australia
judy.kay@sydney.edu.au

ABSTRACT

There has been an explosion of digital learning sensors. Some are in bespoke learning applications. But many more are in the digital tools people use in every aspect of their lives. This paper introduces a user-centred view of EDM for lifelong, life-wide learning. That includes formal education, but goes beyond it to the complex, multi-faceted and ill-defined learning in our broader lives. This is the learning that takes decades and is critical for aspects as diverse as health and wellness, responsible citizenship or working effectively with other people.

The paper begins by asking who the users for EDM are, what their different needs are, and why the answers matter. It then reviews a series of case studies for learning group-work skills. These illustrate the analysis that follows. This starts with the issues for personal data sensing for learning over the long term, in many contexts and aspects of life. Then it considers middleware, a topic rarely discussed in EDM research. Finally, it considers the all important user interfaces for: user control; human-in-the-loop EDM; and learning, particularly, self-monitoring, reflection, planning and broader metacognitive activity. This paper takes a highly critical assessment of over 20 years of my research, from the perspective of user-centred EDM. Building upon that critique, it summarises major mistakes made and lessons learnt and then presents a research agenda and vision.

JEDM Track Journal Papers

(abstracts)

Toward Data-Driven Design of Educational Courses: A Feasibility Study

Rakesh Agrawal
Data Insights Laboratories
ragrawal@acm.org

Behzad Golshan
Boston University
behzad@cs.bu.edu

Evangelos Papalexakis
Carnegie Mellon University
epapalex@cs.cmu.edu

ABSTRACT

A study plan is the choice of concepts and the organization and sequencing of the concepts to be covered in an educational course. While a good study plan is essential for the success of any course offering, the design of study plans currently remains largely a manual task. We present a novel data-driven method, which given a list of concepts can automatically propose candidate plans to cover all the concepts. Our method uses Wikipedia as an external source of knowledge to both identify which concepts should be studied together and how students should move from one group of concepts to another. For our experimental validation, we synthesize study plan for a course defined by a list of concept names from high school physics. Our user study with domain experts finds that our method is able to produce a study plan of high quality.

Next-Term Student Performance Prediction: A Recommender Systems Approach*

Mack Sweeney
Computer Science
George Mason University
Fairfax, VA, USA
msweene2@gmu.edu

Huzefa Rangwala†
Computer Science
George Mason University
Fairfax, VA, USA
rangwala@cs.gmu.edu

Jaime Lester
Higher Education
George Mason University
Fairfax, VA, USA
jlester2@gmu.edu

Aditya Johri
Information Sciences
George Mason University
Fairfax, VA, USA
ajohri3@gmu.edu

ABSTRACT

An enduring issue in higher education is student retention to successful graduation. National statistics indicate that most higher education institutions have four-year degree completion rates around 50%, or just half of their student populations. While there are prediction models which illuminate what factors assist with college student success, interventions that support course selections on a semester-to-semester basis have yet to be deeply understood. To further this goal, we develop a system to predict students' grades in the courses they will enroll in during the next enrollment term by learning patterns from historical transcript data coupled with additional information about students, courses and the instructors teaching them.

We explore a variety of classic and state-of-the-art techniques which have proven effective for recommendation tasks in the e-commerce domain. In our experiments, Factorization Machines (FM), Random Forests (RF), and the Personalized Linear Multiple Regression model achieve the lowest prediction error. Application of a novel feature selection technique is key to the predictive success and interpretability of the FM. By comparing feature importance across populations and across models, we uncover strong connections between instructor characteristics and student performance. We also discover key differences between transfer and non-transfer students. Ultimately we find that a hybrid FM-RF method can be used to accurately predict grades for both new and returning students taking both new and existing courses. Application of these techniques holds promise for student degree planning, instructor interventions, and personalized advising, all of which could improve retention and academic performance.

Acknowledgments

This research was funded by NSF IIS grant 1447489.

*(Does NOT produce the permission block, copyright information nor page numbering). For use with ACM_PROC_ARTICLE-SP.CLS. Supported by ACM.

†Corresponding Author.

Exploring the Effect of Student Confusion in Massive Open Online Courses

Diyi Yang♣, Robert Kraut♢ and Carolyn P. Rosé♣
♣Language Technologies Institute, ♢Human-Computer Interaction Institute
Carnegie Mellon University. 5000 Forbes Ave, Pittsburgh, PA, 15213
{diyiy, kraut, cprose}@cs.cmu.edu

ABSTRACT

Although thousands of students enroll in Massive Open Online Courses (MOOCs) for learning and self-improvement, many get confused, harming learning and increasing dropout rates. In this paper, we quantify these effects in two large MOOCs. We first describe how we automatically estimate students' confusion by looking at their behavior clicking on course content and participating in the course discussion forums. We then apply survival analysis to quantify the impact of confusion on students' dropout. The results demonstrate that the more confusion students express themselves and the more they are exposed to other students' confusion, the sooner they drop out of the course. We also explore the effects of confusion expressed in different contexts and related to different aspects of courses. We conclude with implications for the design of interventions to improve student retention in MOOCs.

Invited Paper

Enabling people to harness and control EDM for lifelong, life-wide learning

Judy Kay
Human Centred Technology Research Cluster
The University of Sydney
Australia
judy.kay@sydney.edu.au

ABSTRACT

There has been an explosion of digital *sensors* of learning. Some are in bespoke learning applications. But many more are in every aspect of our lives. This paper takes a *human-centred view of EDM* for lifelong, life-wide learning, where EDM enables people to harness that data. This view includes formal education. But it goes beyond that, to the complex, multi-faceted, ill-defined and long-term learning needed to work towards the most important goals in our broader lives. Some of the most important of these goals are lifelong and are critical for aspects as diverse as health and wellness, responsible citizenship or working effectively with other people.

The paper begins by considering the stakeholders for EDM. It then presents three longitudinal case studies from my research on supporting learning of complex skills. Drawing on these, the analysis that follows presents lessons learnt and a wish list and vision for future EDM directions towards human-centred EDM.

Keywords

Lifelong learning, life-wide learning, educational data mining, user control, student modelling, learner modelling, personal data, privacy, provenance, user control, business analytics, learning analytics, personal informatics.

1. INTRODUCTION

Technology that can support learning is now pervasive. This has created an explosion of opportunities for life-wide and lifelong learning. One important part of that is formal learning and that has dominated decades of AIED and ITS research. Even this happens in diverse contexts, from physical classrooms to nearly every other place. But it goes well beyond formal learning. The pervasiveness of technology means that we have a rich digital ecosystems. This includes devices that we wear and carry as well as those located and embedded in our environments. From an EDM perspective, this provides many streams of digital footprints that might be harvested to support learning. The potential these offer has created the EDM community [9] as well as many others, such as learning analytics [53] business analytics [37] and personal informatics [38].

Figure 1 illustrates one way to think about EDM as the transformer of *sensor data* into the *learner models* that can support learning. The top of the figure shows a person called Mykola[1] who uses many digital tools and devices. For example, Mykola may interact with a collection of maths tutoring applications, perhaps over many months or even years, using various desktops, tablets and smart phones. Viewing each of these applications as sensors that collect data about Mykola, the figure distinguishes two classes of EDM transformations. At the left are the transformations that feed into the learner model for this *individual learner*. At the right are those that aggregate the data from *collections of learners*. The thinner grey lines, numbered 1 to 4, are classic EDM, interpreting raw sensor data to add it to a learner model of the individual learner (1) and then reasoning on it (2) and the corresponding actions for aggregate models for a collection of learners (3 and 4). Of course the whole point of EDM is to create learner models that are to be used. So the lines 1 and 3 are bidirectional. This reflects the times that the learner model serves information to an application. Creating individual and aggregate learner models may draw on diverse methods and infer aspects that include the learner's knowledge, as well as other critical factors, such as motivation [36].

Over the many years of Mykola's maths education, current EDM approaches produce many learner models. This is what we have seen in over 25 years of AIED research. The learner model has typically been just one part of just one application. It has often been short-lived, perhaps just the single session of a research study. That is changing in two important ways. Learner models are becoming *first class citizens* and they are becoming *long-term*.

One driver for learner models to transition to first class citizen-ship follows from our understanding of their direct value for learners when there is a *user interface* onto them. This has long been called an Open Learner Model [12] and, more recently, similar interfaces are called learning analytics and dashboards [53]. At a quite different level, learner mod-

[1]Thanks to Mykola Pechenizkiy, President of International Educational Data Mining Society for agreeing to be everyman in this image.

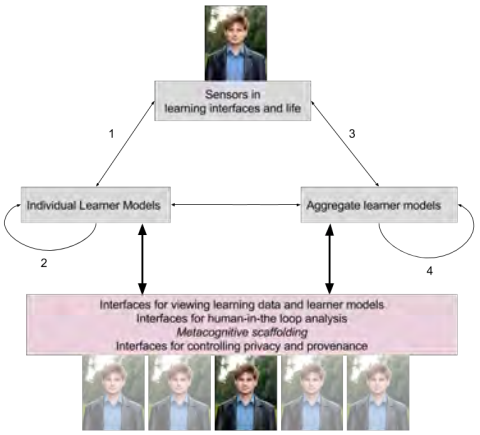


Figure 1: EDM as transformer: from sensor data to learner models as first class citizens

els should have independent standing so that an *application-program interface, API* enables multiple applications to reuse the same learner model. This is a quite different form of openness that is potentially valuable for long term learning. Both these aspects of openness should be designed from a human-centred foundation.

Consider Mykola's broader learning, in areas such as health and wellness. He might use various tracking devices and coaching applications over the long term. The trackers can capture data about aspects like his food intake, physical activity, weight, rest pulse and stress. All of these could serve as sensors for a steadily growing and rich learner model. His interactions with various coaching systems could provide data to model aspects like his metacognitive skills, as he uses each coach to set goals, self-reflect and self-monitor. For life-long goals for good health, Mykola's learner model needs to be kept over the long term, accumulating and mining data from many sensors. Equally, a long-term model representing stable traits, such personality, could be re-used by multiple personalised learning applications [15].

Mykola's sensor data and associated learner model is his *personal data*. People typically want to be able to control such data and its use, as reflected in privacy legislation [34]. User concerns and needs around privacy are important and complex. EDM's learner models may represent aspects that are quite sensitive. These include stable attributes like personality as well ephemeral, sensed attributes such as attention [17]. The aggregated learner model at the right side of Figure 1 has been at the core of EDM research. It is well understood that Mykola's data in these models needs to be treated with care. This is typically achieved with forms of de-identification so that the data actually kept is divorced from the user.

Let us now turn to the lower part of the figure. This depicts several people, in several roles, interacting with suitable interfaces onto these learner models. Some important stakeholders include:

1. builders of ITS/AIED systems;
2. the individual learner;
3. the classroom teacher, parent, mentor, peer learner or other supporter for the learner;
4. administrators, governments and learning "accountants".
5. learning scientists;

In the box above the people, the figure shows *four classes of user interfaces* onto learner models. The first, already mentioned, is the Open Learner Model or learning dashboard. Appropriate forms of these are useful for each stakeholder group. The system builders need them to help understand and debug their systems. Appropriately designed interfaces onto individual and aggregate learner models can be used as part of the *learning process* by the next three groups. The individual learner and their support team are particularly concerned with the individual learner model, although aggregate models can put this in context and help these stakeholders interpret it. The Learning Analytics community has introduced such interfaces for stakeholders at the institutional level with dashboards for administrators. These have much in common with dashboards used in many fields [56, 21]. Learning scientists have a quite different perspective. With their psychology focus, they can use interfaces onto aggregate models to display how people learn, with the exemplar being an increasingly refined understanding of forgetting curves [6].

The figure also highlights other roles. There is an under-explored EDM role for human-in-the-loop systems. Certainly system builders do some of this when debugging. OLMs and dashboards have provided simple interaction to support this role for the learning process stakeholders. The third class of interface shown in the figure is the metacognitive scaffolding that is needed for individual learners because self-reflection, self-monitoring and planning are hard for many people. Finally, the figure show a class of interfaces that enable people to control their own data and provenance. I return to these after the case studies in the next section.

2. CASE STUDIES

This section overviews three strands of my research. The first aimed to help students learn *group work skills*. These are complex and they require many skills. For example, effective group work relies on communication, including listening skills. Long term group work demands considerable self-regulation so that the individual and the group can plan, monitor and reflect on progress. It involves learning leadership skills, which is important for effective teams [51] The second case study is in *computer supported collaborative learning*, a common feature of classrooms. The third concerns design and management of the curriculum for *long term learning* of generic skills across a university degree. All three cases take a human-centred approach to tackling some of the many parts of the complex puzzle of EDM to support lifelong and life-wide learning.

Long term asynchronous group work

Team work skills are important in many contexts. They are so common in the creation of software that computing degrees have a capstone software engineering project. Typically, students work in small teams over a semester to create a substantial software artefact that meets a clients' needs. It is common practice for such teams to use a platform that supports the management of the code and other files as well as the team processes. (Just a few of the many platforms include GitHub[2], trac[3] and BitBucket[4]). These platforms provide rich sensor data about the group behaviour. This case study involved use of trac. We explored how to harness the data about each team member's use of its core tools, a wiki, the ticket system (called issue tracker in other systems) and the version control system (svn in our case). In terms of Figure 1, these three media were the sensors and we wanted to build a model of each individual in the team and make it available to them and their teachers. We wanted to transform the huge amount of data from the sensors (thousands of actions over the three months of the semester) into an OLM.

Figures 2, 3 and 4 illustrate three key stages in our work to create effective OLM interfaces. We began by creating several presentations that were in the spirit of the interaction diagram shown in Figure 2. These were inspired by social translucence visualisations [19]. Each circle represents one team member (the figure has anonymised the display, removing the names that were near each dot). So, for example one user is represented by the pink dot at 2 o'clock. The lines between the dots indicate how much that pair of people interacted on the wiki. This is based on a measure of each person's contributions to the same page. For example, students were advised to create a page for their weekly meeting minutes, with each team member reviewing this and adding a comment, to indicate agreement or identifying problems. If the whole team did this, we would see connections between every pair. In Figures 2, it is striking that the green dot at 7 o'clock is not connected to any others. Worse yet, the green dot represents the person who was supposed to be the team manager. This diagram highlights a potential problem! In practice, these diagrams proved valuable because problems like this became apparent. So long as they were available early enough, and used appropriately in the teaching (not for assessment), they facilitated valuable discussions within the team and with their teachers.

The interaction diagrams and activity diagrams, one for each of the media (wiki, tickets and svn) were our starting point. These were useful for a single point in time. Our next step was to create an OLM showing the long term model. We particularly wanted to be able to see changes. For example, we needed to see what happened after we had identified and tried to remediate a team problem like that of the manager in Figure 2. Our next step was the Wattle Tree visualisation [31], like the example shown in Figure 3[5]. Each green vertical line is one team member. (There are 6 in the figure.) Each day's wiki activity appears as a yellow circle to the left

[2]https://github.com/
[3]https://trac.edgewall.org/
[4]https://bitbucket.org/
[5]The Australian Wattle Tree has small round balls for flowers and they appear in clusters as in the figure.

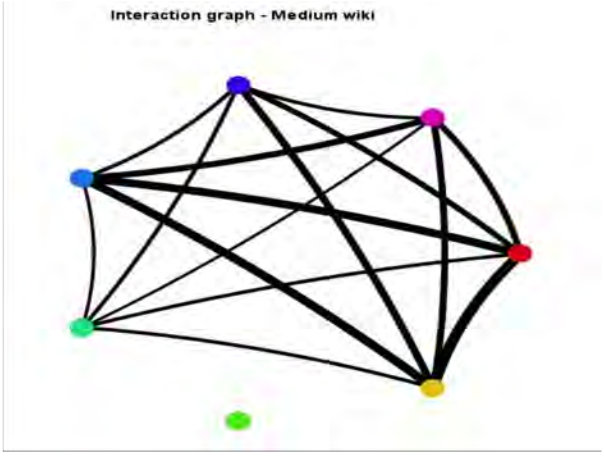


Figure 2: Simple EDM, showing interaction between team members

of the person's line. Each svn action is an orange circle to the right of their line. The size of the circle is a logarithmic function of the count of actions or code lines committed. The green lines are ticket actions. Defining a new task is a dark green line on the left. Completing an allocated task gives the light green ones at the right. This figure shows a high functioning team, with very active leadership by the person on the second line. The rightmost user is clearly less active. The almost barren area towards the top was the semester break. This group was doing well enough that they took a break at that time. This OLM proved useful in the hands of a skillful teacher and team leader. But it was fragile in that weak teams did not use it very well without considerable mentoring. It was also flawed as an interface; the idea of the wattle tree was cute, but it was somewhat forced, difficult to extend and the metaphor does not match the learning goals well.

Our next step was the Narcissus interface [54], shown in Figure 4. Now each user is a block (5 of them in them in the figure). As the legend at the upper right indicates, these show the three media as squares, coloured purple (wiki), blue (svn) and green (tickets). The brighter each square, the more that user did that day on that medium. The bottom of each block shows a cumulative picture for that individual, compared to the team average (grey). Narcissus used quite simple EDM measures, pure counts of actions. But it added scrutability, with the user able to configure how the simple sensor counts map to the colour intensity.

Even more valuable, each cell is interactive. In the figure, the user has clicked a blue (svn) cell and the details are available at the right. This lists details of, and links to, all the changesets to the code checked in by that user on that day. So we now transform the OLM into a navigation tool. This was invaluable for tracking down the details as needed.

A parallel stream of this work explored more sophisticated EDM methods [32, 50] for a cohort of 43 students in 7 groups. The sensor data was 1.6 megabytes in mySQL format, with over 15000 events. With a combination of ex-

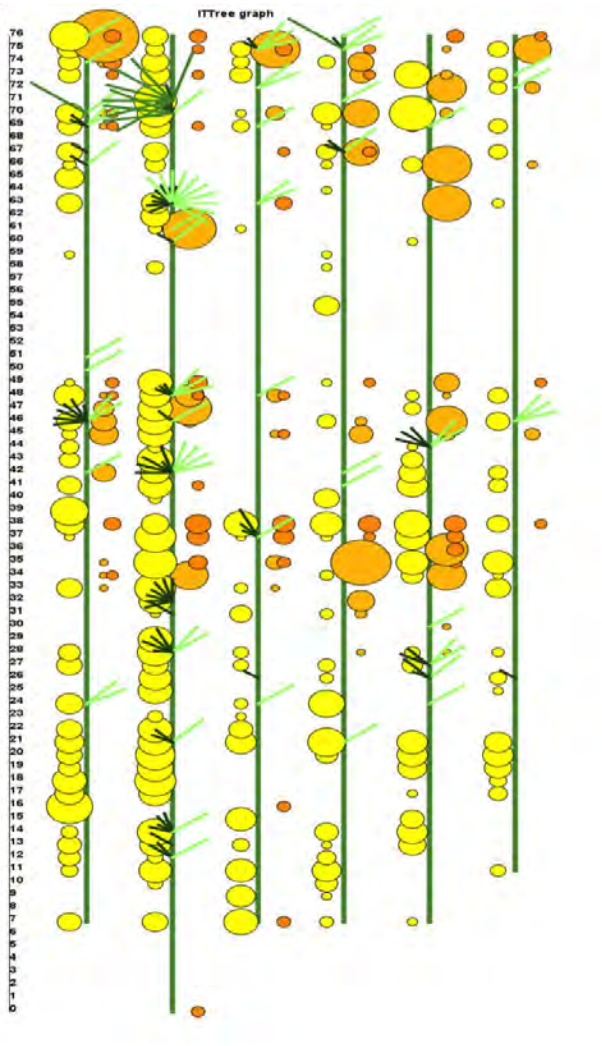


Figure 3: Simple EDM, in a Wattle Tree, showing daily actions by each user on each medium

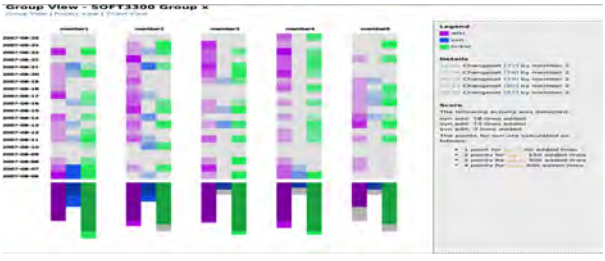


Figure 4: Simple EDM, with Narcissus, a much better and interactive interface onto daily actions by each user on each medium

ploratory and theory-driven approaches, we discovered useful measures of aspects of group and individual behaviour. For example, we identified patterns that are associated with effective leadership. Whether a group's nominated leader exhibited these was predictive of the effectiveness of the leadership. For example, one nominated leader failed to exhibit those patterns, but rather had the pattern of a pure developer, with others having some of the leadership patterns. In fact, the leader was trying to do much of the programming alone, neglecting group management. Other team members tried to fill the leadership void. Promisingly, the group-role patterns were established early in the semester. This is important for remediation. These results also gave the teachers guidance on aspects to teach students and indications of how to better guide them. However, we never translated this more complex approach into an OLM, like Narcissus.

So what did we learn from creating and using the OLMs? They relied on very simple mining of the raw sensor data, just counting actions. And that worked well. In the case of the wiki, actions on the same page were interpreted to model interaction and we used even simpler counts of all ticket actions and lines of code committed. With Narcissus, we transformed this simple data mining into an interactive OLM that supported navigation of the complex information space and this was heavily used in meetings between the tutor and each team and in helping groups with problems. We contributed the Narcissus code to the open source trac project's Track Hacks and used it for several years (until we stopped using trac). We know anecdotally that our students asked for Narcissus to be installed for use in other classes using trac. But there ends it deployment. We are not aware of any team programming environment that provides comparable facilities to model group health. We fell short of any deployment of the complex data mining approaches. It would be excellent to see them integrated into a future version of Narcissus-like tools. That will require design of suitable interfaces. It will be challenging to do this and maintain the Narcissus philosophy of user control. In summary, this sequence of work explored how to harness the digital footprints of teams using trac-like tools and demonstrated the power of simple measures in OLMs and the promise of more sophisticated mining.

Collocated collaborative learning

This case study continues the exploration of how to harness data about groups to inform learning. But we now move to a collocated context (rather than the above asynchronous long term collaboration). This work was inspired by the potential of interactive tabletops to provide a new way to support small group work. This is because small groups commonly work at a table, and tabletops offer a shared interactive canvas. From an EDM perspective, tabletops offer new sensors of collocated collaboration.

Figure 5 shows users in one of our lab studies. This group of three students is doing a collaborative learning task. Each of them had previously worked individually to create a concept map summarising their understanding of provided learning material. Then they came together to create a joint map. We had considerable sensor data: the tabletop touches, so we could determine which learner did each action; the speech captured by the microphone that is visible at the front mid-

Figure 5: Lab study of small group creating a concept map collaboratively

Figure 6: Group health sophisticated dashboard

dle of the table; the initial individual maps, the final group map and all the intermediate states.

To harness all that data, we used data mining to build a model of the quality of the collaboration [45, 44]. Figure 6 shows a set of visualisations for these. The dial at the left summarises how well the group is collaborating [41]. This is left of centre, indicating somewhat poor collaboration. The transformation of sensor data to produce this single measure are complex. They use a model learnt from measures derived from video analysis and the automatically collected sensor data. The other two figures are versions of the interaction and activity diagrams from our earlier work. The middle one shows interactions, such as one user working with interface elements created by another user. The right one shows the level of activity for each user in terms of tabletop touches and speech with both of these highest for the yellow user near 12 o'clock. The left collaboration dial is inscrutable; we never attempted to create an explanation or justification of it for use by a teacher. However, the two simpler diagrams do help as they show elements that use some of the same sensors as that measure.

The tabletop lab work seemed promising. So we then explored how to move it to an authentic classroom, such as shown in Figure 7 [43, 42]. Under the real time and curriculum pressures of a classroom, the teacher wanted to harness the tabletops and an OLM to track each class group's progress on the *content learning objectives* for the class, rather than our inscrutable models of collaboration. This led to the design of the dashboard shown in Figure 8 [41]. The tabletop activity data was used to create models of each group. The OLM for these can be seen at the right of the dashboard. This has a set of bar chars, each group in a different colour, with one bar for each student. This shows a count of the number of propositions that student has created. The darker part of the bar shows propositions matching ones the teacher had defined. Other propositions that students created are indicated in the light parts of each bar. Under the pressure of classroom teaching, even such simple OLMs were more effective when there were alerts to help the teacher see which group seemed to most need attention [42].

Figure 7: Classroom of interactive tabletops

Figure 8: Simple dashboard

So what did we learn? We showed that sophisticated EDM resulted in a model of how collaborative a group was. This black box result has the potential to be valuable information for a classroom teacher. When they need to spend time with one group, they cannot be closely following all of others. So this measure might be helpful when they finish working with one group and need to decide which group most needs their attention. In practice, when we moved to a real classroom, the teachers saw a greater need for simpler measures about the learning activity. The EDM could both provide these in an OLM and drive the alerts to help them quickly decide how to split their time between the groups.

Modelling generic skills over 3-5 years

This case study was driven by two key demands. The first was to improve the design of a university curriculum, with

Figure 9: Aggregate long term learning model of generic skills and potential sensors

a particular focus on systematic building of generic skills, such as group work and communication. A second driver was the needs of accreditation of engineering and IT degree programmes. A human-centred perspective was essential. This is because a whole university degree involves hundreds of academics. We needed all of them to contribute to the project by using our system effectively. More importantly, they control the actual classroom teaching.

We created CUSP [25]. This supports definition of the ontology for the generic skills in the whole of a university degree that runs from 3 to 5 years. CUSP also supports mapping the core ontology, from the university (and faculty) learning outcomes, to each of multiple accreditation standards. It provides views of the curriculum, in terms of these ontologies. CUSP has been in use for several years at the University of Sydney. Indeed, although it was initially designed for use by academics to manage the curriculum design and accreditation processes, it has been repurposed for use by students. This means it used daily, with thousands of students relying on it in Sydney[6].

From the perspective of Figure 1, CUSP provides the *ontology* for an aggregate learner model. The interfaces enable an individual teacher to see their subject in terms of this ontology. The academics responsible for a whole degree have interfaces that show the big picture. For example, Figure 9 shows the seven broad classes of generic skills that are assessed at each of the five levels. This defines the *intended* learning outcomes and level of each across the degree.

Key to the success of CUSP was its mapping of the generic skill ontologies, from the institutional one, as in the figure, to several used by external accreditation bodies. This was based on a very simple approach. There is a mapping only between these ontologies. Each subject co-ordinator maps their detailed learning objectives against the institutional ones. The approach assumes this will correctly translate to the various accreditation ontologies. In practice, our evaluations indicated this worked well [22, 25]. When we examined the small proportion of errors, most were due to the lecturers incorrectly coding against the institutional ontology.

CUSP's ontology is hand crafted centrally with individual teachers mapping their subject against it. This worked well for generic attributes. We explored how to take this approach further, to deal with subject matter content. We did this in the context of the Programming Fundamentals sequence of subjects in Computer Science. These are intended to build and develop skills and knowledge, with students reaching higher levels of mastery over several subjects and several years. The result was ProGoSs [23]. This enabled teachers to map either their subject description of their exam against a standard curriculum, such as ACM 2013. Our educators found exam mapping easier. ProGoSs provided a framework, or ontology, for a learner model. Notably, teachers needed to augment the standard concepts. This is partly because standard curricula need to be framed in general terms. By contrast, an actual subject learning objectives are linked to aspects like the particular programming language, as well as very fine-grained or new concepts. The actual exam became the sensor. The marks for each question were added to the individual learner models. This detailed learner modelling has considerable potential for EDM or learner analytics if combined with other information about the individual learner. ProGoss was tested over multiple subjects in multiple institutions. But it has not been in broader use.

A similar approach is being incorporated into tools like Gradescope[7]. The demands of accreditation, linked to the process of grading exams, seem potential drivers. The human factors will be critical for real world deployment. These include interfaces that make it easy to create the learner model and it will rely of the value of the learner models for the stakeholders.

In both CUSP and ProGoSs, the main stakeholders were the custodians of the curriculum, both at the level of the degree and the individual subject. CUSP has been repurposed for student use because its curriculum role meant that it encoded considerable detail of each subject and how it fits into the degree programme. ProGoSs, as a research prototype, foreshadows the potential for tracking fine-grained learning progression.

3. LESSONS LEARNT, FEARS, VISION

The stakeholders identified in the introduction all share the need for EDM to provide evidence-based insights about learners. But there are important differences. In terms of Figure 1, the individual learner model and the aggregate models have different roles. By contrast, the learning scientists are concerned with the aggregate models only. For all the stakeholders aiding the individual learner, that learner's individual learner model is key. But these stakeholders also need an aggregate model, to make sense of the individual one. For example, for a learner to judge their progress, they may want to compare the learning model against those of successful students – where some students want to compare against bare pass students and others against the high achievers.

The builders of ITS/AIED systems need EDM to drive personalised learning. Learning scientists want to understand learning more broadly. Both these roles reflect core goals of AIED/ITS communities, from their foundations. Both are well represented in current EDM research and in deployed AIED systems.

[6]and thousands more use a commercial version called U-Improve http://www.u-planner.com/products/u-improve

[7]https://gradescope.com/

But EDM can also give a quite different level of benefit. One of these is the OLM or dashboard interface into a learner model. As in Figure 1, EDM can be seen as any process that transforms data from sensor of learning into a learner model, where this is any representation of learners that is intended to support learning. Then a suitable OLM (or dashboard) has the potential to serve many purposes [12], listed below an illustrated in terms of Narcisses.

- Improving the accuracy of the model − students could set the counts that triggered colour changes;
- Supporting metacognitive processes of planning, monitoring and reflection, with evidence supporting self-awareness [52] − the group's tutor played a key role in useing the OLM to support individuals and the group.
- Facilitating collaboration or competition − the main teachers used the OLM in meetings with the managers from each group so each manager could use the OLM to share their current challenges and actions;
- Facilitating navigation − with the OLM linking to each sensor element;
- Respecting the learner's right to access and control their personal data, and their trust in the learner model − all sensor data was accessible and controlled by the students;
- Using the learner model as an assessment of the learner − this was purely formative and was far too simplistic to use directly to assess the learning objectives.

A human-centred view frames EDM in terms of the needs of stakeholders. One such need is for interfaces onto the learner models. This should also impact the design of sensors, the way that they are processed, the design of learner models, the ways that sensors contribute to them and the ways that people can control and harness their own data. Taking this perspective, what are the key lessons from the case studies?

Embrace simplicity, with care

Baker has suggested that we need "stupid tutoring systems, and intelligent humans" [8] where this is possible because of rich data-driven teaching system. This matches our experiences. A human-centred perspective favours at least taking serious account of the simplest approaches to the design of each element of the EDM processes, as in Figure 1. All three case studies indicated simple models were valuable and could be deployed.

EDM can point to the need create new sensors. For example, we concluded that we needed to link the tabletop touches to the individual who did the touch. Tabletop hardware generally does not support this (one exception being DiamondTouch). To create this sensor, we integrated a Kinect with the tabletop to provide a hardware independent way. When EDM uses systems that were not designed for it, we may need new sensors to support downstream simplicity of the EDM. There is a need for this in MOOCs. These appear to have been created without EDM as a core design driver. So it is hard to link each sensor, for example video activity and MCQ responses, to learning objectives for learner modelling [33].

CUSP has proved useful for curriculum management and its very simple mapping of learning "ontologies" has worked well in practice. ProGoSs has potential that has yet to be demonstrated in a deployed system. For long term learner models, our work with CUSP, ProGoSs and infrastructures highlight the power of exploring simple approaches. Even these demand effort to create effective interfaces and to carefully take account for the pragmatics that will mean that people will see it as worth investing the time needed to make use of them.

In terms of Figure 1, *simple interfaces* onto *simple learner models* have huge but under-exploited potential. The human, context, cultural and interface challenges are critical and should underpin EDM design. There is a risk in taking too narrow a focus or too simplistic approaches that miss these [40]. Standardised tests and arbitrary data within administrative systems are simple sensors to drive EDM to produce aggregate models. But they also pose potential risks for misuse, for example, creating pressure for teachers and learners to focus on improving arbitrary measures on whatever is readily tested even if these relate only weakly to important learning outcomes.

Gaming-aware design

Gaming educational systems occurs when people subvert or violate the use that was intended and is required to support effecyive learning. We can expect gaming [3]. If we intend EDM for real world use, beyond the lab, we need to consider the drivers and opportunities for gaming by any and all of the stakeholders. The direct sensor data and the learner models of EDM are supremely "gamable" at many levels.

High stakes, simplistic use of learning data invites gaming. Even quite low stakes system, such as an ITS with formative assessment can have gaming [7] and there may be a fine line between gaming and "help" [1]. ITS/AIED/EDM communities have amassed considerable experience of gaming both in recognising it and using that to tackle it. This could be a foundation for a checklist to help system builders inform design, by considering potential gaming throughout the EDM process.

We detected gaming with Narcissus. For example, a student was called to account in one class for their lack of recent visible activity. In the following week, they appeared to have considerable activity. The design of Narcissus, its use, meant the group mentor simply clicked on each link to each action, to navigate through the activity. When that revealed trivial actions to create the *appearance* of activity, it created a teachable moment! Narcissus's simple measures, and its scrutability, meant that students see how to game it if its use is simple-minded. At the same time, its direct mapping to navigation of the complex space of the trac site made it easy for a teacher to scrutinise the actual activity.

One might expect that gaming would not occur in systems for personal use alone. There would seem to be no point in gaming the system. The learner would simply be fooling

themselves, reflected in OLMs that are incorrect. Our earliest work aimed to provide a personal learning tool about a text editor [14]. Even so, we discovered one (and only one) user who tediously used the OLM interface to indicate they knew a great deal. Their log of actual activity told a very different story.

Our Personis learner model representation was robust against this form of gaming. It kept all evidence from its sensors. These were: log data use of the editor, with inferences translating this into a model of demonstrated knowledge; data from the use of the OLM. The OLM used a *resolvers* to interpret the evidence from the learner as more reliable than the usage analysis. Other resolvers treated behaviour as more reliable. Perhaps the OLM interface should have helped the learner appreciate this.

Personal data-mining

It is currently difficult for people to mine their control data. This is true on every level. It is often challenging to *access* the data, be it at the sensor level or within a learner model. Beyond that, it is difficult for a typical use to combine sources of data and to analyse collections in useful ways. Figure 1 shows interfaces for human-in-the-loop analyses such as advocated for machine learning generally [2]. These approaches seem particularly important for personal data mining so that the individual can annotate and preprocess their own data, its interpretation and processing.

Reflecting its roots, EDM has substantial work in sophisticated tutoring systems. Issues of personal data mining may be less pressing in these. But for the many sources of simple sensor data, the Quantified Self community is already exploring how to mine many forms of their own data, typically to gain self-awareness and often to tackle important long term goals. The famous example of Nicholas Felton [18] points to the huge amounts of diverse data that an individual can amass and then actually harness for their own needs. Another example is lifelogging, for example based on worn cameras [39] to collect rich personal data, requiring sophisticated image processing to support personal data mining.

When personal data mining is conducted by a learner, it has the potential to play a key role in the metacognitive and self-regulation processes. These are already central to the use of data by the Quantified Selfers. We know that many learners benefit from metacognitive scaffolding for such activities [5]. We need to explore how to create these.

It is unsurprising that mainstream data mining deals with big data, and aggregate model for many people. For example, Microsoft has a patent on personal data mining [48] but this is actually about mining of personal-data. EDM researchers are perfectly placed to lead initiatives in the quite different task of personal data-mining. It calls for methods that can deliver useful insights with the relatively small amounts of data of an individual.

Infrastructures for personal data-mining

Figure 1 hints at the infrastructures needed for the EDM processes of lifelong and life-wide learning. The sensors it shows are already embedded in the many technologies we use across our lives. These include formal learning with its plethora of devices and applications, ranging from the LMSs to the thousands to specialised apps, videos and other learning researches that can produce digital footprints.

Currently, the sensor data is splattered across many devices that we own as well as in the cloud and managed by diverse services that we do not control. When we explored how people wanted such data managed [11], we learnt that most people want to have control over it, even when they cannot yet establish a use for it. If we are to make use of such findings, we need to explore how to create infrastuctures that can support people in bringing together their diverse personal data. We will also need to tackle the HCI challenges of creating interfaces that enable people to actually manage their collected data and use it effectively.

One strand of my research has explored how to create an infrastructure for a lifelong learner model [30]. This is similar to the vision recently proposed by Nye [47] where a learner model becomes a web-service. My Personis family of learner models can make flexible use of ubicomp sensors [13, 4]. A central design goal was to support user control. So its representation has hooks that interface designers need to create interfaces that enable people to control data *from the sensors*, *in the learner model*, including the reasoning *within the model* and in all *uses of the model* by applications [29].

The infrastructure needed for lifelong EDM is similar to the notion of a personal data vault [46]. The key difference is that a learner model is more than a unified collection of sensor data. It needs to be designed to answer questions that matter to learners, either with direct access to the data, via an OLM interface, or indirectly because it can be used effectively in one or more applications. This was a driver for work, like CUSP and ProGoSs to help define the learner model ontology, and mappings between multiple ontologies [47]. In the case of CUSP, the ontologies were handcrafted. But our ProGoSs work explored use of standard and widely used resources, such as international and professional curriculum specifications for the ontology of learning elements. For the levels of learning, ProGoSs imported the definition and associated tutoring elements to help classroom teachers understand them. Our experiments with Bloom and Neo-Piagetian learning taxonomies make it clear that teachers can to learn to use these effectively [24]. This will enrich progressive modelling of lifelong learning.

EDM has made real headway on this problem of infrastructures for *aggregate models*. For example, the DataShop [35] has tackled infrastructure issues, providing standards for representation, tools for analysis and interfaces. These are valuable for the builders of ITS/AIED systems and for learning scientists. Learners may well be willing to contribute their learner models to similar aggregate data stores in a form of citizen science, so long as they have the assurances they need about the management of their data, including provenance metadata and privacy [20].

Interfaces for user control

System builders could consider how their design decision impact user control. This is partly a matter of taking the perspective of the learner, or other users, when building el-

ements for EDM. For example, how can a learner define the structure of their long term learner model? And then link in the sensors? For the case of personal informatics sensors, such as physical trackers, we have demonstrated that a promising approach is to create interfaces for people to define goals [10] and link various sensors to these. In our user study, people could readily think about this data and its use in terms of their goals. They could then link the goals to various sensors, such as a FitBit, mobile phone app or a smart cushion. This is one human-centred approach to the design of control interfaces for infrastructures for EDM.

At a quite different level, *scrutability* could become a criterion for the design of the EDM, as well as software architectures [28]. These could then flow into a test-driven approach for building EDM systems. For this, we need to identify success measures that include interpretability of the EDM processes and learner models [49].

Stealth assessment [55] is appealing since it enables a learner to focus on learning, and getting assessment measures for free. These approaches could be made compatible with user control if the user is able to define what the systems that can log, as has been done for computer use [27].

Conclusions

Figure 1 presented a view of EDM with sensors associated with each learner, and EDM processes transform the sensor data into learner models. This view highlights the individual learner model, which holds only the data of one learner. This is increasingly becoming a long term *first class citizen*, independent of any one application, especially for reuse by multiple applications [16]. The data within the EDM system belongs to the learner (even if they may barely be aware of that). An OLM can enable a learner to see and, perhaps, also make good use of it. This personal EDM is catching on in the Quantified Self community.

A human-centred view of EDM will raise the profile of all the interfaces in the figure. One of these will scaffold metacognitive processes, to help the learner make effective use of their OLM. A quite different class of interfaces is needed to ensure learners can manage their learner models. It will be far harder to graft these on, as an after-thought, to the EDM processes. As we design and build each element of the EDM processes, we need to consider this goal. For example, this calls for consideration of how intelligible the processes are. We may begin to measure the trade-off between the performance of an EDM algorithm and how easy it is to explain it in ways different stakeholders find satisfactory. It will require capture of provenance and support for people to use this to define how they want their learner model used.

EDM and ITS/AIED have built strong foundations for aggregate learner modelling. These fit well into our historic core business of building personalised teaching systems, that combine aggregate learner models with the individual learner's model. Aggregate models could be key EDM contributions to learning science. They are also core for Learning Analytics, especially for use by teachers and the administrators.

Returning to the three cases studies, all relate to learning complex skills that we develop over years. All involved *lab* and *deployed* systems. All explored both *simple* and *sophisticated* EDM. The longitudinal group work had rather *conventional sensors*, based on interactions with trac. The tabletops involved more *diverse sensors*. The effectiveness of that deployed EDM in the classroom relied heavily on the ways that *students* and *teachers* used the OLMs. The tabletop work was driven by the needs of *classroom teachers*, with student interfaces still on the future work slate. CUSP and PRoGoS explored *infrastructures* for long term learning, with key stakeholders being the *curriculum caretakers*, both *administrators* at the level of the whole degree and the *teachers* of individual subjects.

This paper has presented a reflection on three strands of my research, with lesson learnt and how they might contribute to a vision for EDM research. In terms of the sensors, the learner models and the stakeholders, these case studies are outside the mainstream of EDM. This seems set to change. For example, Baker has comments that: "there is a disconnect between the vision of what intelligent tutoring systems could be, and what they are ... between the most impressive examples of what intelligent tutors can do, and what current systems used at scale do". He highlights the power achieved by a human-centred approach, with extensive EDM informing the design and refinement of the ASSISTments system [26]. One of the key lessons of the three case studies is that we have much to gain from *simplicity*. It is important for practical and useful deployments of systems. It should help in the design of interfaces for scrutability. This paper has argued for the need for a set of evidence-based guidelines for EDM design that considers the *many levels of gaming*. These must particularly help account for potential misuse of learner models when a stakeholder group repurposes them. I have proposed that we explore *personal data mining*, potentially building links with the Quantified Self movement. Finally, it calls for EDM research into practicalities of creating *infrastuctures for EDM* with associated interfaces so *people can control their data and associated EDM processes*. These are parts of a broad vision for EDM that supports lifelong and life-wide learning.

4. ACKNOWLEDGMENTS

The research discussed in this paper was supported by several sources: Australian Research Council; Smart Internet Co-operative Research Centre; Smart Services Co-operative Research Centre; the Faculty Research Cluster Program, Faculty of Engineering and Information Technologies, The University of Sydney.

5. REFERENCES

[1] V. Aleven, I. Roll, B. M. McLaren, and K. R. Koedinger. Help helps, but only so much: Research on help seeking with intelligent tutoring systems. *International Journal of Artificial Intelligence in Education*, pages 1–19, 2016.

[2] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza. Power to the people: The role of humans in interactive machine learning. American Association for Artificial Intelligence, 2014.

[3] D. Ariely and S. Jones. *Predictably irrational*. HarperCollins New York, 2008.

[4] M. Assad, D. Carmichael, J. Kay, and B. Kummerfeld.

Personisad: Distributed, active, scrutable model framework for context-aware services. *Pervasive Computing*, pages 55–72, 2007.

[5] R. Azevedo and A. F. Hadwin. Scaffolding self-regulated learning and metacognition–implications for the design of computer-based scaffolds. *Instructional Science*, 33(5):367–379, 2005.

[6] A. D. Baddeley. *Human memory: Theory and practice*. Psychology Press, 1997.

[7] R. Baker, J. Walonoski, N. Heffernan, I. Roll, A. Corbett, and K. Koedinger. Why students engage in "gaming the system" behavior in interactive learning environments. *Journal of Interactive Learning Research*, 19(2):185, 2008.

[8] R. S. Baker. Stupid tutoring systems, intelligent humans. *International Journal of Artificial Intelligence in Education*, 26(2):600–614, 2016.

[9] R. S. Baker and K. Yacef. The state of educational data mining in 2009: A review and future visions. *JEDM-Journal of Educational Data Mining*, 1(1):3–17, 2009.

[10] D. Barua, J. Kay, B. Kummerfeld, and C. Paris. Modelling long term goals. In *User Modeling, Adaptation, and Personalization*, pages 1–12. Springer, 2014.

[11] D. Barua, J. Kay, and C. Paris. Viewing and controlling personal sensor data: what do users want? In *Persuasive Technology*, pages 15–26. Springer, 2013.

[12] S. Bull and J. Kay. Smili: a framework for interfaces to learning data in open learner models, learning analytics and related fields. *International Journal of Artificial Intelligence in Education*, pages 1–39, 2016.

[13] D. J. Carmichael, J. Kay, and B. Kummerfeld. Consistent modelling of users, devices and sensors in a ubiquitous computing environment. *User Modeling and User-Adapted Interaction*, 15(3-4):197–234, 2005.

[14] R. Cook and J. Kay. The justified user model: a viewable, explained user model. In *Proceedings of the Fourth International Conference on User Modeling UM94*, pages 145 – 150, 1994.

[15] M. Dennis, J. Masthoff, and C. Mellish. Adapting progress feedback and emotional support to learner personality. *International Journal of Artificial Intelligence in Education*, pages 1–55, 2015.

[16] P. Dillenbourg. The evolution of research on digital education. *International Journal of Artificial Intelligence in Education*, 26(2):544–560, 2016.

[17] S. K. D'Mello. Giving eyesight to the blind: Towards attention-aware AIED. *International Journal of Artificial Intelligence in Education*, 26(2):645–659, 2016.

[18] C. Elsden, D. S. Kirk, and A. C. Durrant. A quantified past: Toward design for remembering with personal informatics. *Human–Computer Interaction*, pages 1–40, 2015.

[19] T. Erickson and W. A. Kellogg. Social translucence: an approach to designing systems that support social processes. *ACM transactions on computer-human interaction (TOCHI)*, 7(1):59–83, 2000.

[20] A. Fekete, J. Kay, M. Franklin, D. Barua, and B. Kummerfeld. Managing information for personal goals (vision). In *Data Engineering Workshops (ICDEW), 2015 31st IEEE International Conference on*, pages 30–33. IEEE, 2015.

[21] S. Few. *Information dashboard design*. O'Reilly, 2006.

[22] R. Gluga, J. Kay, and T. Lever. Modeling long term learning of generic skills. In *Intelligent Tutoring Systems*, pages 85–94. Springer, 2010.

[23] R. Gluga, J. Kay, R. Lister, M. Charleston, J. Harland, D. Teague, et al. A conceptual model for reflecting on expected learning vs. demonstrated student performance. In *Proceedings of the Fifteenth Australasian Computing Education Conference-Volume 136*, pages 77–86. Australian Computer Society, Inc., 2013.

[24] R. Gluga, J. Kay, R. Lister, and S. Kleitman. Mastering cognitive development theory in computer science education. *Computer Science Education*, 23(1):24–57, 2013.

[25] R. Gluga, J. A. Kay, and T. Lever. Foundations for modeling university curricula in terms of multiple learning goal sets. *Learning Technologies, IEEE Transactions on*, 6(1):25–37, 2013.

[26] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.

[27] Z. Hinbarji, R. Albatal, N. O'Connor, and C. Gurrin. Loggerman, a comprehensive logging and visualization tool to capture computer usage. In *MultiMedia Modeling*, pages 342–347. Springer, 2016.

[28] J. Kay. Scrutable adaptation: Because we can and must. In *Adaptive hypermedia and adaptive web-based systems*, pages 11–19. Springer, 2006.

[29] J. Kay and B. Kummerfeld. Creating personalized systems that people can scrutinize and control: Drivers, principles and experience. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(4):24, 2012.

[30] J. Kay and B. Kummerfeld. Lifelong learner modeling. *Adaptive Technologies for Training and Education*, pages 140–164, 2012.

[31] J. Kay, N. Maisonneuve, K. Yacef, and P. Reimann. The big five and visualisations of team work activity. In *Intelligent tutoring systems*, pages 197–206. Springer, 2006.

[32] J. Kay, N. Maisonneuve, K. Yacef, and O. Zaïane. Mining patterns of events in students' teamwork data. In *Proceedings of the Workshop on Educational Data Mining at the 8th International Conference on Intelligent Tutoring Systems (ITS 2006)*, pages 45–52, 2006.

[33] J. Kay, P. Reimann, E. Diebold, and B. Kummerfeld. Moocs: So many learners, so much potential... *IEEE Intelligent Systems*, (3):70–77, 2013.

[34] A. Kobsa. Privacy-enhanced web personalization. In *The adaptive web*, pages 628–670. Springer, 2007.

[35] K. R. Koedinger, R. S. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the edm community: The pslc datashop. *Handbook of educational data mining*, 43, 2010.

[36] K. R. Koedinger, E. Brunskill, R. S. Baker, E. A. McLaughlin, and J. Stamper. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, 34(3):27–41, 2013.

[37] R. Kohavi, N. J. Rothleder, and E. Simoudis. Emerging trends in business analytics. *Communications of the ACM*, 45(8):45–48, 2002.

[38] I. Li, Y. Medynskiy, J. Froehlich, and J. Larsen. Personal informatics in practice: improving quality of life through data. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, pages 2799–2802. ACM, 2012.

[39] N. Li, M. Crane, C. Gurrin, and H. J. Ruskin. Finding motifs in large personal lifelogs. In *Proceedings of the 7th Augmented Human International Conference 2016*, AH '16, pages 9:1–9:8, New York, NY, USA, 2016. ACM.

[40] D. Y.-T. Liu, T. Rogers, and A. Pardo. Learning analytics - are we at risk of missing the point? In *Proceedings of the 32nd Ascilite Conference*, 2015.

[41] R. M. Maldonado, J. Kay, K. Yacef, and B. Schwendimann. An interactive teacher's dashboard for monitoring groups in a multi-tabletop learning environment. In *Intelligent Tutoring Systems*, pages 482–492. Springer, 2012.

[42] R. Martinez-Maldonado, A. Clayphan, and J. Kay. Deploying and visualising teacher's scripts of small group activities in a multi-surface classroom ecology: a study in-the-wild. *Computer Supported Cooperative Work (CSCW)*, 24(2-3):177–221, 2015.

[43] R. Martinez Maldonado, Y. Dimitriadis, J. Kay, K. Yacef, and M.-T. Edbauer. Orchestrating a multi-tabletop classroom: from activity design to enactment and reflection. In *Proceedings of the 2012 ACM international conference on Interactive tabletops and surfaces*, pages 119–128. ACM, 2012.

[44] R. Martinez-Maldonado, Y. Dimitriadis, A. Martinez-Monés, J. Kay, and K. Yacef. Capturing and analyzing verbal and physical collaborative learning interactions at an enriched interactive tabletop. *International Journal of Computer-Supported Collaborative Learning*, 8(4):455–485, 2013.

[45] R. Martinez-Maldonado, K. Yacef, and J. Kay. Data mining in the classroom: Discovering groupsâĂŹ strategies at a multi-tabletop environment. In *Proceedings of the International Conference on Educational Data Mining*, pages 121–128, 2013.

[46] M. Y. Mun, D. H. Kim, K. Shilton, D. Estrin, M. Hansen, and R. Govindan. Pdvloc: A personal data vault for controlled location data sharing. *ACM Transactions on Sensor Networks (TOSN)*, 10(4):58, 2014.

[47] B. D. Nye. Its, the end of the world as we know it: Transitioning aied into a service-oriented ecosystem. *International Journal of Artificial Intelligence in Education*, 26(2):756–770, 2016.

[48] R. Ozzie, W. Gates, G. Flake, T. Bergstraesser, A. Blinn, C. Brumme, L. Cheng, M. Connolly, N. Dani, D. Glasgow, et al. Personal data mining, 2011. US Patent 7,930,197.

[49] R. Pelánek. Metrics for evaluation of student models. *JEDM-Journal of Educational Data Mining*, 7(2):1–19, 2015.

[50] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaïane. Clustering and sequential pattern mining of online collaborative learning data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(6):759–772, 2009.

[51] E. Salas, D. E. Sims, and C. S. Burke. Is there a "Big Five" in Teamwork? *Small Group Research*, 36(5):555–599, Oct. 2005.

[52] D. A. Schön. *The reflective practitioner: How professionals think in action*, volume 5126. Basic books, 1983.

[53] G. Siemens and R. S. d Baker. Learning analytics and educational data mining: towards communication and collaboration. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, pages 252–254. ACM, 2012.

[54] K. Upton and J. Kay. Narcissus: group and individual models to support small group work. In *User modeling, adaptation, and personalization*, pages 54–65. Springer, 2009.

[55] L. Wang, V. Shute, and G. R. Moore. Lessons learned and best practices of stealth assessment. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)*, 7(4):66–87, 2015.

[56] O. M. Yigitbasioglu and O. Velcu. A review of dashboards in performance management: Implications for design and research. *International Journal of Accounting Information Systems*, 13(1):41–59, 2012.

Full Papers

{ENTER}ing the Time Series {SPACE}: Uncovering the Writing Process through Keystroke Analyses

Laura K. Allen[1], Matthew E. Jacovina[1], Mihai Dascalu[2], Rod D. Roscoe[1], Kevin M. Kent[1], Aaron D. Likens[1], & Danielle S. McNamara[1]
[1]Arizona State University, Tempe, USA
[2]University Politechnica of Bucharest, Bucharest, Romania
{LauraKAllen, Matthew.Jacovina}@asu.edu, {Mihai.Dascalu}@cs.pub.ro, {Rod.Roscoe, Kkent4, Aaron.Likens, dsmcnama}@asu.edu

ABSTRACT
This study investigates how and whether information about students' writing can be recovered from basic behavioral data extracted during their sessions in an intelligent tutoring system for writing. We calculate basic and time-sensitive keystroke indices based on log files of keys pressed during students' writing sessions. A corpus of prompt-based essays was collected from 126 undergraduates along with keystrokes logged during the session. Holistic scores and linguistic properties of these essays were then automatically calculated using natural language processing tools. Results indicated that keystroke indices accounted for 76% of the variance in essay quality and up to 38% of the variance in the linguistic characteristics. Overall, these results suggest that keystroke analyses can help to recover crucial information about writing, which may ultimately help to improve student models in computer-based learning environments.

Keywords
Intelligent Tutoring Systems; Writing; Natural Language Processing; Feedback; Keystrokes; Temporality

1. INTRODUCTION
Effective written communication is a complex socio-cognitive skill that is important for success in academic and professional settings [1-2]. The writing process relies on both lower- and higher-level knowledge and skills, ranging from knowledge of the language and domain to strategies necessary for generating inferences and flexibly adapting to different task demands [1; 3-5] Not surprisingly, then, the development of strong writing skills is extremely difficult and students consistently underachieve on national and international assessments of writing [6-8].

The remediation of these writing deficits is a similarly challenging task. The development of writing proficiency demands that students have access to high-quality instruction that is attuned to their particular needs. Research on writing instruction finds that students attain the greatest benefits when they are provided strategy instruction, practice, and feedback [9-10]. In particular, *deliberate practice* is crucial for the development of writing skills [11] and has been shown to help students regulate the planning, drafting, and reviewing stages of writing [10]. This type of meaningful and mindful practice inherently relies upon individualized formative feedback—feedback that reveals and explains actionable steps that students must take to improve. However, in large classrooms, detailed and targeted feedback on multiple essay drafts per student presents a daunting challenge for teachers.

Computer-based tools such as automated writing evaluation (AWE) systems have been developed to alleviate some of the pressures facing writing instructors [12]. At their core, AWE tools implement natural language processing (NLP) and machine learning techniques to accurately model the scores that expert human raters would assign based on the structure and content of students' essays [13-14]. Additionally, many AWE systems and intelligent tutoring systems (ITSs) incorporate instructional elements such as lessons and practice games [15-16]. These modern systems extend beyond the assessment of essay quality to provide students with personalized feedback and recommendations for improvement.

Although a wealth of research has been conducted to validate the *accuracy* of AWE scores, much less attention has been paid to the pedagogical and rhetorical elements of these systems. Specifically, critics often cite the lack of sensitivity to different audiences, rhetorical moves, and writing processes as serious areas of concern, which can lead to impersonal and ineffective instruction and feedback [17;18]. These critiques are valid and point to much needed future research. Accordingly, researchers and developers have begun to re-focus their efforts away from establishing the accuracy of scoring models and towards the improvement of the personalized and nuanced aspects of the feedback and instruction.

To better detect and respond to differences among students' writing processes and behaviors, we may need to embed assessments that are based on more than their written products and essay scores. These measures can be either visible or hidden from users (i.e., "stealth assessments") [19], and can inform specific instruction and feedback that is tailored to students' individual habits. In the context of computer-based learning environments, these assessments can be informed by a wealth of information that is easily logged within the system. Snow and colleagues (2014) [20], for example, developed stealth assessments of self-regulation within a reading comprehension tutoring system. They found that the predictability of students' choices in the system was

indicative of their self-regulation skill and influenced their performance on the learning task. Overall, such assessments may offer a viable solution to the writing process assessment problem. Both simple measures (e.g., typing speed) and complex measures (e.g., trajectories of mouse movements) might allow us to model the writing processes and characteristics of student users.

In this paper, we examine the efficacy of behavioral measures that are accessible (but rarely collected or analyzed) in writing training systems to detect information about students' performance on their essays. In particular, we examine whether basic and time-sensitive keystroke indices can be used to model the scores and linguistic features of students' essays. Our ultimate goal is to use these models to provide more individualized tutoring and feedback to students.

1.1 Keystroke Analyses for Writing

Keystroke data presents a potentially valuable approach for modeling students' writing behaviors [e.g., 21]. Although researchers have made significant strides in leveraging the linguistic features of texts to understand writing quality, there has been substantially less research on students' online or real-time writing processes. Due to challenges of data collection, prior writing research has focused primarily on students' finished writing products and not their moment-by-moment writing processes. Recently, however, keystroke logging tools (i.e., software that records the keys individuals press while typing) have been applied to the study of writing [22]. These tools offer a viable way to study students' actions as they compose and edit their essays. One such tool, InputLog, has been developed to interface with NLP tools, which enables analyses that synthesize both keystroke and linguistic data.

Illustrative examples of the value of keystroke analyses stem from work on affect detection during writing [21; 23]. Writers' affective states during writing—ranging from boredom and frustration to excitement and engagement—can have a significant impact on the writing experience and eventual products. However, these qualities may not be detectable from written products alone. How might keystroke patterns vary when writers are in a fluid, engaged "flow" state as compared to a frustrated struggle to generate ideas?

In recent work, Bixler and D'Mello (2013) [21] have begun to explore such questions. They collected individual difference measures and keystroke data from student writers to detect online affective states during writing (i.e., self-reported affective states in 15-second intervals). Their results indicated that a combination of behavioral (keystroke) measures and student-level indices was able to detect boredom, engagement, and neutral states between 11% and 38% above baseline. Similarly, Allen et al. (in press) [23] combined individual difference, linguistic, and keystroke indices to predict engagement and boredom across writing sessions. Their results suggested that these three categories of indices were successful in modeling students' affective states during writing. Indices related to academic ability, text properties, and keystroke logs were able classify high and low engagement and boredom in writing sessions with 77% accuracy.

In sum, keystroke analyses hold the potential to reveal crucial data on students' online writing experiences and processes that are normally invisible in product-based analyses alone.

1.2 Writing Pal

A long-term goal of our research is to improve personalized, adaptive learning and feedback within the Writing Pal (W-Pal) intelligent tutoring system [24]. W-Pal offers explicit strategy instruction, practice, and feedback for prompt-based persuasive essay writing for high school and early college students. Relative to other writing training systems (see [24] for a review), W-Pal is unique in its focus on explicit strategy instruction and its varied opportunities for practice (i.e., game-based strategy practice and essay writing practice). Strategy instruction is delivered via video presentations on canonical writing processes: prewriting, drafting, and revising. These videos feature virtual pedagogical agents who explain and demonstrate a variety of principles and strategies (see Figure 1 for a screenshot of the Freewriting Module). These lessons include: Freewriting and Planning (prewriting); Introduction Building, Body Building, and Conclusion Building (drafting); and Paraphrasing, Cohesion Building, and Revising (revising). After completing lessons, students unlock a suite of strategy practice mini-games. In these games, students reinforce their strategy knowledge through both generative and identification tasks. Game-based practice allows students to work on specific components of the writing process and strategies prior to applying them in a complete essay composition.

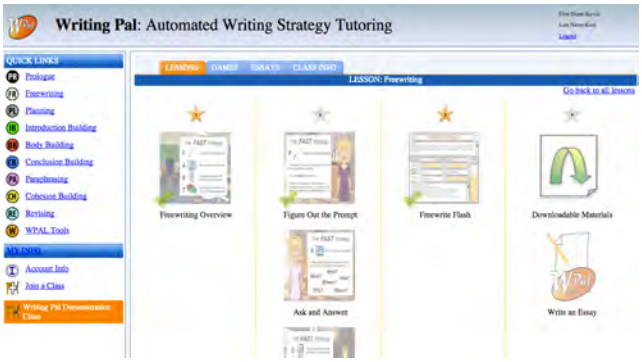


Figure 1. Screenshot of the the Freewriting module

1.2.1 W-Pal Essay Practice and Feedback

W-Pal also gives students the opportunity to practice writing persuasive essays and receive summative and formative feedback. Writing takes place in a word-processing interface where students can view the prompt, a "scratch-pad" for brainstorming and outlining, and the writing space. Once the essays are submitted, a combination of formative and summative feedback is provided. Like other AWEs, W-Pal employs NLP tools to extract linguistic data from essays, and implements a series of algorithms to assess quality and guide feedback delivery. In analyzing the text, the system considers characteristics across a variety of linguistic indices.

Summative feedback (see Figure 2) includes a holistic score on a 1-6 scale, with descriptors representing each level (i.e. "Great"). Formative feedback (see Figure 2) is given both at the essay-level (i.e. length, relevance, structure) and section-level (i.e. suggestions to improve an introduction). This formative feedback is designed to be specific, actionable, and aligned to strategies taught in the lessons. For example, students who submit essays with weak conclusions may receive feedback about summarizing key arguments from the body paragraphs in the conclusion. After viewing the feedback, students can revise their essays. In the

revision phase, essay feedback is displayed adjacent to the writing space, facilitating uptake of the recommendations.

Previous research evaluating the efficacy of the W-Pal system has found that this training results in improved essay scores, increased strategy knowledge, and improved revising strategies [15; 25-26].

Figure 2. Screenshot of the feedback window

1.3 CURRENT STUDY

The current study investigates how and whether information about students' writing behaviors within W-Pal can be recovered from basic behavioral data extracted from keystroke analyses. To this end, we calculate a number of indices based on the keystrokes pressed by student writers with the intent of modeling the quality and linguistic features of their essays. An overarching aim of this research is to develop online, stealth assessments of students' writing processes that can inform new student models and system adaptivity. An increase in the sensitivity of W-Pal to students' writing processes is expected to improve its ability to offer more nuanced and personalized feedback and recommendations.

We collected timed, persuasive essays written by undergraduate students and scored using the W-Pal algorithm [27]. Linguistic properties of the essays were assessed via Coh-Metrix [28] and WAT [29], which are automated NLP tools that calculates text information related to lexical, syntactic, cohesive, and rhetorical properties. In addition, we logged keystrokes during students' writing session and calculated measures related to the general and temporal properties of these keystroke logs.

We hypothesized that these basic and time-sensitive keystroke indices would provide meaningful information about the writing processes enacted by students, which would subsequently relate to the quality and characteristics of their essays.

2. METHODS

2.1 Participants

We recruited 131 undergraduate participants from a university in the United States, who received course credit. Students reported a mean age of 19.8 years, with 44.3% identifying as female, 64.1% Caucasian, 14.5% Asian, 7.6% African American, 7.6% Hispanic, and 6.1% as "Other." Data for five students were lost due to computer error; thus, the final corpus comprised 126 essays.

2.2 Data Collection Procedure

Participants wrote a timed (25-minute), prompt-based, persuasive essay. Essay prompts resembled typical SAT items, and students were not allowed to proceed until the full 25 minutes elapsed. Students typed their essays in the AWE component of W-Pal and all keystrokes were logged along with millisecond timestamps. Essays contained an average of 412.3 words (*SD* = 159.9, *min* = 47.0, *max* = 980.0).

2.3 Essay Scoring

Students' essays were automatically scored using a computational algorithm that assigns scores on a scale from 1 (lowest) to 6 (highest). This algorithm relied on linguistic features computed by Coh-Metrix, the Writing Assessment Tool (WAT), and Linguistic Inquiry and Word Count (LIWC). For more details on this algorithm, see [27].

2.4 Text Analyses

Linguistic properties of essays were assessed via two NLP tools: Coh-Metrix [28] and WAT [29]. These tools report hundreds of linguistic indices that relate to text structure, general readability, rhetorical patterns, lexical choices, and cohesion. For the current analyses, we selected four indices from Coh-Metrix and WAT that demonstrated theoretical ties to writing quality. We chose this limited number of indices to specifically examine whether and how the keystroke indices would map onto four key dimensions of the essays: lexical, syntactic, semantic, and cohesion.

Word Frequency. Coh-Metrix and WAT calculate multiple indices that describe the specific types of words used in texts. Word frequency measures, for instance, are used to assess how frequently certain words occur in the English language. Coh-Metrix reports indices of word frequency that are taken from the CELEX database. Additionally, Coh-Metrix reports the logarithm of word frequency for all words in a text. An index of log frequency is calculated because reading times are typically linearly related to the logarithm of word frequency rather than the raw word frequency [30]. For this reason, we chose to examine the log frequency of all words.

Syntactic Complexity. Additionally, Coh-Metrix and WAT contain a number of indices that describe the properties of the sentences in texts, such as the frequency of specific parts of speech and the complexity of their syntactic constructions. Sentence complexity is assessed by multiple indices. More complex syntax is typically associated with higher quality essays [28] and recent evidence suggests that working memory capacity is linked to the production of more complex syntax [31]. Here, we used the index mean number of words before the main verb as a proxy for sentence complexity.

Semantic Diversity. Semantic diversity refers to the number of unique concepts expressed in an essay. This measure is conceptually similar to measures of lexical diversity, but more strongly emphasizes the diversity of ideas rather than specific words. A semantic diversity score is calculated in WAT using Latent Semantic Analysis (LSA) [32] and is operationalized as the ratio of semantically independent concepts to the total number of word types in an essay.

Global Semantic Cohesion. Global semantic cohesion is also calculated in WAT using LSA. Here, we used the index LSA (start-to-end), which calculates the degree to which the introduction and conclusion of an essay contain semantically similar information. We chose this index (rather than examining the semantic similarity between all the paragraphs) because higher-quality essays typically share semantic content in the opening and closing paragraphs, but bring in outside information in the form of arguments and evidence in the body paragraphs.

3. KEYSTROKE ANALYSES

To investigate whether and how students' writing behaviors were related to the quality and linguistic properties of their essays, we computed a number of keystroke indices. In particular, we calculated both *basic keystroke indices* (i.e., indices that were aggregated across the entire essay), as well as *time-sensitive keystroke indices* (i.e., indices that accounted for the temporal nature of the keystroke data).

3.1 Basic Keystroke Indices

Basic keystroke indices aggregated the number of specific writing events (e.g., pauses and backspaces) that occurred across an entire writing session. These basic indices are deliberate replications of indices from previous studies because they have been successfully used to model students' affect during writing [21; 23]. Table 1 provides an overview of these indices.

Table 1. Basic Keystroke Indices

Measure	Description
Verbosity	Number of keystrokes per essay
Backspaces	Number of backspaces per essay
Largest Latency	Largest time difference between keystrokes during essay writing
Smallest Latency	Smallest time difference between keystrokes during essay writing
Median Latency	Median of all the differences in time between keystrokes per essay (not including initial pause)
Initial Pause	Length of the first pause of an essay writing session
0.5 Second Pauses	Number of pauses above .5 seconds and below 1 second
1 Second Pauses	Number of pauses above 1 second and below 1.5 seconds
1.5 Second Pauses	Number of pauses above 1.5 seconds and below 2 seconds
2 Second Pauses	Number of pauses above 2 seconds and below 3 seconds
3 Second Pauses	Number of pauses above 3 seconds

3.2 Time-Sensitive Keystroke Indices

Despite the importance of basic keystroke indices, indices that aggregate behavioral patterns over the course of an entire essay session can miss out on important temporal variability. For instance, consider the time series depicted in Figure 3. This plot shows the number of keystrokes pressed by one student writer within each 30 second window of a writing session. The student clearly did not maintain stable behavioral patterns throughout the writing session; instead, she engaged in periods of high and low activity. Analyses that are restricted to basic indices necessarily ignore this variability. We hypothesize that investigations into the temporal structure of the keystrokes (i.e., the distributions of events in time) will provide meaningful information about students' writing processes beyond the basic aggregated measures.

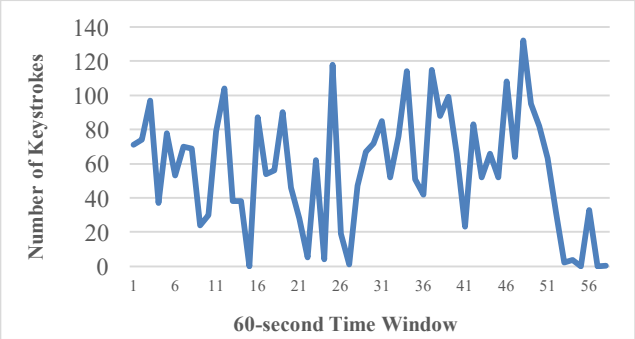


Figure 3. Variability of keystroke patterns for a single student

Table 2. Time-Sensitive Keystroke Indices

	Description
StDev Events	Standard deviation of the number of events in each time window
Slope Degree	Slope of the linear regression applied on the time series
Entropy	Shannon's Entropy calculated for the number of events in the windows normalized by the total number of events for the overall time series. If a student only typed in a single window, the entropy would be 0. When maintaining a constant typing rate, entropy converges toward the maximum value of log(n).
Degree of Uniformity	Uniformity of the time series (Jensen-Shannon divergence method), which is a symmetric and bounded function of similarity that calculates the similarity between two distributions: a uniform probability distribution of 1/n (i.e., a constant typing rate) and the probability of key presses in a given window (i.e., the actual time series produced by the student).
Local Extremes	Number of time windows for which the direction of the evolution of keystroke events changes. This reflects inconsistency in writing rates across the windows.
Average Recurrence	Average recurrence of events across the time windows. This recurrence is expressed as the distances between time windows that contain at least one keystroke event. This measure is useful for identifying writing pauses. If each time window has at least one event, recurrence is 0, whereas if students take long pauses that occasionally result in time windows of 0 events, recurrence increases (if they write every two time windows, recurrence will be one).
StdDev Recurrence	Standard deviation of the recurrence across the time windows

Note: All time-sensitive keystroke indices were calculated using 30- and 60-second time windows.

To this end, we calculated a number of new indices that we have classified as *time-sensitive keystroke indices*. These indices deliberately take the within-subject temporal distribution of keystroke events into account. The time series of keystrokes

generated during students' sessions were first separated into non-overlapping windows of 30 and 60 seconds to account for variability across different scales. These individual windows contained information about the number of keystroke events that occurred in each time window. The time-sensitive keystroke indices were then separately generated based on each of the two window intervals (see Table 2).

3.3 Statistical Analyses

Statistical analyses investigated whether basic and time-sensitive keystroke indices accounted for variability in student writing performance. Pearson correlations were first calculated between the holistic essay scores and the keystroke indices obtained from the writing sessions (see Tables 1 and 2). Indices that displayed a significant or marginally significant correlation with essay scores ($p < .10$) were retained in the analysis.

Normality of the indices was assessed with skew, kurtosis, and visual data inspections, and no indices were removed based on these inspections. Range transformations (0-1) were applied to ensure that the keystroke and linguistic indices were on the same scale. Multicollinearity was then assessed among the indices ($r > .90$). When two or more indices demonstrated multicollinearity, the index that correlated most strongly with essay scores was retained in the analysis.

A linear regression analysis[1] was conducted using M5-prime feature selection to assess which of the remaining keystroke indices were most predictive of essay scores. To avoid overfitting the model, we chose a ratio of 15 essays to 1 predictor, which allowed for a maximum of eight indices to be entered in to the model, given that there were 126 essays included in the analysis.

We first conducted the regression analysis on the entire corpus, and then validated the model using ten-fold cross-validation with shuffled sampling. In this cross validation analysis, the corpus was first split into 10 "folds" and each fold was individually removed from the corpus for each analysis and the remaining essays were used as the training set. We tested the accuracy of the linear regression model by examining its ability to model the omitted fold. The process was repeated until each fold was omitted once in the test set. This analysis therefore allowed us to test the model's accuracy on independent sets of data (i.e., data that are not in the training set). If the overall model and the model generated by the cross-validation analysis are similar, our confidence in model stability is increased.

Following this essay score analysis, similar follow-up analyses were conducted using the keystroke indices to predict the linguistic features of the essays. For these analyses, we followed the same procedure detailed above.

4. RESULTS

4.1 Keystrokes and Essay Quality

Pearson correlations were calculated between the basic and time-sensitive keystroke indices and students' holistic essay scores to examine the strength of the relationships among the variables. The correlation analysis revealed that there were 10 keystroke indices that demonstrated a significant relation with holistic essay scores and did not demonstrate multicollinearity with each other. To avoid overfitting the model, we only selected the eight indices that were most strongly correlated with essay scores. These eight indices are listed in Table 3.

Table 3. Correlations between Essay Scores and Keystroke Indices

Keystroke Index	*r*	*p*
Verbosity	0.819	<.001
Local Extremes (30s time window)	-0.476	<.001
Entropy (30s time window)	0.472	<.001
Median Latency	-0.436	<.001
StdDev Events (30s time window)	0.397	<.001
Largest Latency	-0.359	<.001
Backspaces	0.308	<.001
StdDev Recurrence (30s time window)	-0.297	= .001

A linear regression analysis was calculated with the eight keystroke indices as predictors of students' essay scores (score range: 1-6). This analysis yielded a significant model, $R^2 = .758$, $RMSE = 0.377$, $p < .001$, with three variables that combined to account for 76% of the variance in the essay scores: *Verbosity* [$\beta = 1.03$, $p < .001$], *Largest Latency* [$\beta = -.09$, $p < .001$], and *Backspaces* [$\beta = .39$, $p < .001$]. The follow-up ten-fold cross validation analysis produced a significant model with similar statistics, $R^2 = .737$, $RMSE = 0.386$.

An interesting question is whether additional indices provided useful information about the essay quality once Verbosity was removed from the analysis. That is, including the total number of key presses may suppress the important role of other writing behaviors. We conducted a second regression analysis that excluded Verbosity. This regression yielded a significant model, $r = .778$, $R^2 = .606$, $RMSE = 0.482$, $p < .001$. Six variables were significant or marginally significant predictors in the regression analysis and combined to account for 61% of the variance in students' essay scores: *StdDev Events* (30s) [$\beta = 0.529$, $p < .001$], *Entropy* (30s) [$\beta = 1.047$, $p < .001$], *StdDev Recurrence* (30s) [$\beta = -0.509$, $p < .001$], *Backspaces* [$\beta = 0.209$, $p < .01$], *Local Extremes* (30s) [$\beta = -0.176$, $p < .05$], and *Median Latency* [$\beta = -0.141$, $p = .096$]. As above, the cross validation model produced similar results, $R^2 = .588$, $RMSE = 0.534$.

In sum, these correlation and regression analyses indicate that better writers pressed more keys (both characters and backspace) over the course of their writing session. They also maintained a more consistent rate across the 30 second time windows (i.e., whether they typed or not within the individual time windows), as measured by Entropy, Local Extremes, and StdDev Recurrence indices, but exhibited greater variability in the *number of keystroke events* within the 30s time windows (StdDev Events). Additionally, these students' keystroke logs were characterized by shorter pause times as measured both by the Median and Largest Latency indices. Taken together, these findings demonstrate that *writing fluency*—the ease and consistency with which writers generate text—is a key indicator of proficiency (e.g., [33]). This work both confirms and extends prior research by investigating a

[1] We investigated the usefulness of a number of regression and neural net techniques in the current study. However, due to space limitations, these models are not reported. In the end, we report the the linear regression models because this approach yielded the strongest and most stable models.

feature of higher quality writing using process analyses rather than post-hoc linguistic analyses alone.

4.2 Keystrokes and Linguistic Features

Our second aim was to investigate whether keystroke indices were related to specific linguistic features of the essays. Pearson correlations were calculated between the keystroke indices and the four linguistic variables calculated by Coh-Metrix and WAT. These analyses were then followed by a regression analysis, and validated using ten-fold cross validation. The statistical information for these resulting models is provided below.

Word Frequency. The word frequency regression analysis yielded a significant model, R^2 = .185, *RMSE* = 0.179, $p < .001$. Three variables were significant or marginally significant predictors: *2 Second Pauses* [β = -0.278, $p < .01$], *Initial Pause* [β = 0.203, $p < .05$], and *0.5 Second Pauses* [β = 0.208, $p = .06$]. The cross validation model was significant, R^2 = .204, *RMSE* = 0.187.

Syntactic Complexity. None of the keystroke indices were significantly or marginally significantly correlated with the selected measure of syntactic complexity.

Semantic Diversity. The analysis to predict the semantic diversity in essays yielded a significant model, R^2 = .375, *RMSE* = 0.123, $p < .001$. Five variables were significant predictors in this regression analysis: *1 Second Pauses* [β = -0.379, $p < .001$], *StdDev Events* (30s) [β = -0.361, $p < .01$], *Slope Degree* (30s) [β = 0.336, $p < .01$], *Median Latency* [β = -0.265, $p < .05$], and *Local Extremes* (60s) [β = 0.173, $p < .05$]. The cross-validation analysis yielded a significant model, R^2 = .255, *RMSE* = 0.133.

Global Semantic Cohesion. Analyses to predict global semantic cohesion based on keystroke data yielded a significant model, R^2 = .194, *RMSE* = 0.238, $p < .001$ with four significant predictors: *StdDev Events* (30s) [β = 0.477, $p < .01$], *3 Second Pauses* [β = 0.424, $p < .001$], *Verbosity* [β = 0.337, $p < .01$], and *Median Latency* [β = 0.307, $p < .05$]. The model produced by the cross-validation analysis was significant, R^2 = .160, *RMSE* = 0.244.

The results of the linguistic analyses indicate that the basic and time-sensitive keystroke indices were meaningfully related to the linguistic features of students' essays at multiple levels. Notably, however, the linguistic regression models were weaker than the essay score model, and the findings were less robust to the cross-validation procedure.

The model generated to predict semantic diversity was the strongest of the linguistic models. This analysis indicated that more semantically diverse essays were related to shorter pauses, with more variability at the 60-second time window (Local Extremes), but less variability at the 30-second time windows. The global semantic cohesion and word familiarity models were also significant with keystroke indices for both accounting for just under 20% of the variance in the linguistic properties. Finally, the syntactic complexity measure was not significantly related to any of the keystroke indices, indicating that perhaps behavioral patterns do not manifest in the different sentence structures produced by writers.

5. DISCUSSION

AWE systems provide an environment for students to receive writing instruction and engage in deliberate practice with summative and formative feedback [12]. Despite the general success of their scoring algorithms (e.g., [13-14; 27]), however, the pedagogical elements of these systems have much room for improvement. For instance, one major weakness of AWE systems is that they they typically only adapt to student users based on individual essay drafts. System developers tend to rely on NLP methods to examine the quality of students' written products; yet, information about their behavioral processes is largely ignored.

In the current study, we used system logs of keystrokes to develop online assessments of students' writing performance. The behavioral processes enacted by writers are important elements of writing skill [1; 22]; therefore, our aim was to determine whether we could assess and model the quality and linguistic properties of students' essays by calculating indices related to their typing behaviors. Basic and time-sensitive keystroke indices were calculated to analyze the behavioral patterns enacted by student writers. These indices provided information about writing processes at both the aggregate level (e.g., total number of pauses and backspaces) as well as information about how these behaviors unfolded over time. The results revealed that keystroke indices were able to model over three-quarters of the variance in students' essay scores. Additionally, these indices were able to model the linguistic properties of the essays at multiple levels.

The essay score analyses revealed that 10 keystroke indices were significantly correlated with students' holistic essay scores. This is important because it indicates that information about the quality of students' essays can be detected by analyzing their behavioral processes. Further, the two regression analyses revealed that the total number of keystrokes pressed by writers provided the most predictive power in the model, but that without this measure of Verbosity, the remaining indices were still about to account for 61% of the variance in essay scores.

These initial analyses of essay score indicate that *fluency* may be an important skill that is captured by the keystroke indices. In our study, the students who produced higher-quality essays were also more consistent in their typing (i.e., whether they typed or not) across the 30 second time windows, yet they had higher variability in the *number* of keystroke events they produced in these time windows. This finding suggests that these students' writing sessions may have been characterized by short (rather than long) patterns of writing and pausing. Some confirmation for this intuition is found in the the negative correlations between essay score and pause times (i.e., Median and Largest Latency). However, future research will need to examine these writing-pause patterns more closely. It may be the case, for instance, that short pauses are indicative of thoughtful writing, such as the search for appropriate words or phrases rather than "freewriting" behavior. Long pauses, on the other hand, may be indicative of mind wandering that warrants system intervention.

Follow-up linguistic analyses similarly revealed important information about the role of behavioral processes in writing. These analyses first indicated that the basic and time-sensitive keystroke indices were significantly related to the linguistic features of students' essays at the lexical, semantic and global cohesion levels, but not at the syntactic level. This indicates that keystroke indices may be picking up on specific meaning-making processes, rather than differences in cognitive factors, such as working memory capacity. For instance, semantic diversity represents the number of semantically related concepts that appear in students' essays, which may map onto the differences in the content that students chose to include in their essays. Syntactic complexity, on the other hand, is much more weakly related to the

meaning of a particular text and, instead, may be indicative of individual differences in specific cognitive skills (e.g., [31; 34]).

It is important to note that the keystroke indices accounted for a smaller amount of the variance in linguistic properties than in the overall essay scores. This suggests that variations in students' behavioral patterns may manifest in the properties of students' essays in different ways depending on the specific context. For instance, long pauses may be more indicative of cohesion if students are writing about an unfamiliar topic that requires more deliberate planning. On the other hand, if students are writing in response to a familiar or emotionally charged topic, it may be the case that essay cohesion will be associated with rapid typing with minimal pauses. The results of these follow-up analyses suggest that future analyses may need to use content-based information to make predictions about the relevance and interpretation of particular keystroke indices. Analytic techniques that allow the system to take past behavior and prompt content into consideration, for instance, could go a long way in improving the interpretability of these patterns.

These results are promising and suggest that keystroke indices can be utilized to uncover important information about the behavior and performance of student writers. Here, we analyzed the keystrokes produced for a short, prompt-based essay task. In the future, additional studies will be conducted to specifically examine how these keystroke patterns map onto writing across different genres, contexts, and difficulty levels. For example, multiple writing sessions could be collected for each participant, with prompt difficulty, genre, or audience varying across these sessions. This research design would help to disentangle signals that vary across multiple factors, such as boredom and difficulty.

Another area for future research lies in the calculation of more sophisticated keystroke indices, as well as the integration of keystroke indices with other system information. We used only keystroke indices as our predictors because we were interested in the degree to which simple behavioral measures alone could predict information about students' essays. In future studies, it will be important to consider additional indices that may be related to the context of these writing behaviors. For instance, if we aim to model students' engagement during writing, it will be important to collect additional information from our systems, such as their prior writing behaviors (e.g., on previous essays, or from original to revised drafts), as well as the linguistic content of the essays.

The overarching goal of this research is to enhance AWE systems such that they provide feedback and instruction that is more attuned to writers' processes. Eventually, we aim to be able to identify specific behavioral patterns associated with different writing processes, which will allow us to provide students with more pointed, online feedback and instruction. For example, through the combination of multiple keystroke indices, systems may be able to distinguish when students are experiencing writer's block as opposed to when they are engaged in the task, but have paused to think. If writer's block were detected, W-Pal could then ask students if they need help or offer specific strategies and practice opportunities for idea generation.

Overall, our results suggest that time-sensitive behavioral data can (and, in our opinion, should!) be used to help drive more personalized feedback and instruction in computer-based learning environments. Although a number of future studies are needed to investigate how this keystroke information can be used most effectively, the current study takes a strong first step in revealing the power of these indices.

6. ACKNOWLEDGEMENTS

The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120707 to Arizona State University. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

7. REFERENCES

[1] Graham, S. 2006. Writing. In P. Alexander & P. Winne (Eds.), *Handbook of educational psychology* (pp. 457-477). Mahwah, NJ: Erlbaum.

[2] National Commission on Writing. 2003. *The Neglected "R."* College Entrance Examination Board, New York.

[3] Allen, L. K., Snow, E. L., Crossley, S. A., Jackson, G. T., and McNamara, D. S. 2014. Reading comprehension components and their relation to the writing process. *L'année psychologique/Topics in Cognitive Psychology, 114*, 663-691.

[4] Allen, L. K., Snow, E. L., and McNamara, D. S. 2016. The narrative waltz: The role of flexibility on writing performance. *Journal of Educational Psychology.* doi: 10.1037/edu0000109

[5] Donovan, C. A., and Smolkin, L. B. 2006. Children's understanding of genre and writing development. In C. A. MacArthur, S. Graham, & J. Fitzgerald (Eds.) *Handbook of writing research* (pp. 131-143). New York: Guilford.

[6] Baer, J. D., and McGrath, D. 2007. *The reading literacy of U.S. fourth-grade students in an international context: Results from the 2001 and 2006 Progress in International Literacy Study (PIRLS)*. National Center for Educational Statistics, Institute of Education Sciences, U.S. Department of Education.

[7] National Assessment of Educational Progress. 2007. *The nation's report card: Writing 2007*. Retrieved Nov. 20, 2010, nces.ed.gov/nationsreportcard/writing/

[8] National Assessment of Educational Progress. 2011. *The nation's report card: Writing 2011*. Retrieved Nov. 5, 2012, nces.ed.gov/nationsreportcard/writing.

[9] Graham, S. and Perin, D. 2007. A meta-analysis of writing instruction for adolescent students. *Journal of Educational Psychology, 99*, 445-476.

[10] Kellogg, R., and Raulerson, B. 2007. Improving the writing skills of college students. *Psychonomic Bulletin and Review, 14*, 237-242.

[11] Johnstone, K. M., Ashbaugh, H., and Warfield, T. D. 2002. Effects of repeated practice and contextual-writing experiences on college students' writing skills. *Journal of Educational Psychology, 94(2)*, 305-315.

[12] Allen, L. K., Jacovina, M. E., and McNamara, D. S. 2016. Computer-based writing instruction. In C. A. MacArthur, S.

Graham, & J. Fitzgerald (Eds.), *Handbook of writing research (2nd ed.)* (pp. 316-329). New York, NY: The Guilford Press.

[13] Dikli, S. 2006. An overview of automated scoring of essays. *Journal of Technology, Learning, and Assessment, 5*.

[14] Shermis, M., and Burstein, J. (Eds.). 2003. *Automated essay scoring: A cross-disciplinary perspective*. Erlbaum, Mahwah, NJ.

[15] Allen, L. K., Crossley, S. A., Snow, E. L., and McNamara, D. S. 2014. Game-based writing strategy tutoring for second language learners: Game enjoyment as a key to engagement. *Language Learning and Technology, 18*, 124-150.

[16] Roscoe, R. D., Allen, L. K., Weston, J. L., Crossley, S. A., and McNamara, D. S. 2014. The Writing Pal intelligent tutoring system: Usability testing and development. *Computers and Composition, 34*, 39-59.

[17] Deane, P. 2013. On the relation between automated essay scoring and modern views of the writing construct. *Assessing Writing, 18*, 7–24.

[18] Perelman, L. 2012. Construct validity, length, score, and time in holistically graded writing assessments: The case against automated essay scoring (AES). In C. Bazerman, C. Dean, J. Early, K. Lunsford, S. Null, P. Rogers, & A. Stansell (Eds.), *International advances in writing research: Cultures, places, measures* (pp. 121–131). Fort Collins, Colorado: WAC Clearinghouse/Anderson, SC: Parlor Press.

[19] Shute, V. J. 2011. Stealth assessment in computer-based games to support learning. In S. Tobias & J. D. Fletcher (Eds.), *Computer games and instruction* (pp. 503-524). Charlotte, NC: Information Age Publishers.

[20] Snow, E. L., Jacovina, M. E., Allen, L. K., Dai, J., and McNamara, D. S. 2014. Entropy: A stealth assessment of agency in learning environments. In J. Stamper, Z. Pardos, M. Mavrikis, & B. M. McLaren (Eds.), *Proceedings of the 7th International Conference on Educational Data Mining*, (pp. 241-244). London, UK.

[21] Bixler, R. and D'Mello, S. 2013. Detecting boredom and engagement during writing with keystroke analysis, task appraisals, and stable traits. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces* (pp. 225-234). New York, NY: ACM.

[22] Leijten, M., and Van Waes, L. 2013. Keystroke logging in writing research: Using Inputlog to analyze and visualize writing processes. *Written Communication, 30*, 358-392.

[23] Allen, L. K., Mills, C., Jacovina, M. E., Crossley, S. A., D'Mello, S. K., and McNamara, D. S. in press. Investigating boredom and engagement during writing using multiple sources of information: The essay, the writer, and keystrokes. In *Proceedings of the 6th International Learning Analytics and Knowledge (LAK) Conference.*

[24] Roscoe, R. D., and McNamara, D. S. 2013. Writing Pal: Feasibility of an intelligent writing strategy tutor in the high school classroom. *Journal of Educational Psychology, 105,* 1010-1025.

[25] Allen, L. K., Crossley, S. A., Snow, E. L., Jacovina, M. E., Perret, C. A., and McNamara, D. S. 2015. Am I wrong or am I right? Gains in monitoring accuracy in an intelligent tutoring system for writing. In A. Mitrovic, F. Verdejo, C. Conati, & N. Heffernan (Eds.), *Proceedings of the 17th International Conference on Artificial Intelligence in Education*. Madrid, Spain.

[26] Roscoe, R. D., Snow, E. L., Allen, L. K., and McNamara, D. S. 2015. Automated detection of essay revising patterns: application for intelligent feedback in a writing tutor. *Technology, Instruction, Cognition, and Learning, 10,* 59-79.

[27] McNamara, D. S., Crossley, S. A., Roscoe, R. D., Allen, L. K., and Dai, J. 2015. Hierarchical classification approach to automated essay scoring. *Assessing Writing, 23*, 35-59.

[28] McNamara, D. S., Graesser, A. C., McCarthy, P., and Cai, Z. 2014. *Automated evaluation of text and discourse with Coh-Metrix*. Cambridge: Cambridge University.

[29] McNamara, D. S., Crossley, S. A., and Roscoe, R. D. 2013. Natural language processing in an intelligent writing strategy tutoring system. *Behavior Research Methods, 45*, 499-515.

[30] Haberlandt, K., and Graesser, A. C. 1985. Component processes in text comprehension and some of their interactions. *Journal of Experimental Psychology, 114,* 357-374.

[31] Allen, L. K., Perret, C., and McNamara, D. S. in press. Linguistic signatures of cognitive processes during writing. Manuscript submitted to the *Annual Cognitive Science (Cog Sci) Society conference*.

[32] Landauer, T., McNamara, D. S., Dennis, S., and Kintsch, W. (Eds.). 2007. *Handbook of Latent Semantic Analysis.* Mahwah, NJ: Erlbaum.

[33] Chenoweth, N. A., and Hayes, J. R. 2001. Fluency in writing generating text in L1 and L2. *Written communication, 18*(1), 80-98.

[34] Kemper, S., Rash, S., Kynette, D., and Norman, S. 1990. Telling stories: The structure of adults' narratives. *European Journal of Cognitive Psychology*, *2*, 205-228.

Automatic Gaze-Based Detection of Mind Wandering during Narrative Film Comprehension

Caitlin Mills[a], Robert Bixler[a], Xinyi Wang, & Sidney K. D'Mello
University of Notre Dame
384 Fitzpatrick Hall, Notre Dame, IN, 46556, USA
[cmills4, rbixler, xwang24, sdmello]@nd.edu

ABSTRACT
Mind wandering (MW) reflects a shift in attention from task-related to task-unrelated thoughts. It is negatively related to performance across a range of tasks, suggesting the importance of detecting and responding to MW in real-time. Currently, there is a paucity of research on MW detection in contexts other than reading. We addressed this gap by using eye gaze to automatically detect MW during narrative film comprehension, an activity that is used across a range of learning environments. In the current study, students self-reported MW as they watched a 32.5-minute commercial film. Students' eye gaze was recorded with an eye tracker. Supervised machine learning models were used to detect MW using global (content-independent), local (content-dependent), and combined global+local features. We achieved a student-independent score (MW F_1) of .45, which reflected a 29% improvement over a chance baseline. Models built using local features were more accurate than the global and combined models. An analysis of diagnostic features revealed that MW primarily manifested as a breakdown in attentional synchrony between eye gaze and visually salient areas of the screen. We consider limitations, applications, and refinements of the MW detector.

Keywords
mind wandering; film comprehension; machine learning; eye gaze

1. INTRODUCTION
Mind wandering (MW) reflects an attentional shift from task-related to task-unrelated thoughts [31]. MW is estimated to consume half of our everyday thoughts [19] and can occur at almost any time – driving down the road, eating a meal, or during a classroom lecture. There are some benefits to our innate ability to MW, specifically with respect to planning and creativity [34]. However, MW has some detrimental effects as well, particularly in the realm of education [30]. A recent meta-analysis across 88 independent samples indicated that MW was negatively correlated with performance, and that the negative relationship was stronger for more complex tasks such as reading comprehension [26]. Given the negative impact of MW on learning [29, 30], it is important to develop attention-aware systems that can reorient attention when MW occurs [8]. However, these systems require reliable MW detection, which is the focus of this work.

MW detection can be particularly challenging since MW is an internal state with few overt markers (unlike some emotions per se). It can even be difficult for people to realize when they are MW, as it can occur without metacognitive awareness [30]. Moreover, the onset and duration of MW cannot be clearly demarcated as with other disengaged behaviors, such as gaming the system or WTF (Without Thinking Fastidiously) behaviors [1, 25].

In the present study, we focus on detecting MW in the novel educational context of narrative film comprehension – a more complex task than self-paced reading where most MW detection efforts have focused on. We chose this task for two reasons. First, a large number of students from all over the world watch educationally relevant films and recorded lectures daily, particularly in the advent of massive open online courses (MOOCs). Second, MW is quite frequent in online video lectures: students report MW around 40% of the time while viewing lectures [29, 33], so there is considerable promise to detecting and responding to MW in this context.

1.1 Background and Related Work
Only one study (to our knowledge) has attempted MW detection while students viewed dynamic visual scenes, such as the narrative film we consider here. Pham and Wang [25] detected MW while students watched video lectures on a smart phone with a MOOC-like application and responded yes or no to thought probes during the lectures. They used student heart rate (extracted via photoplethysmography) to train classifiers to detect MW. They achieved a 22% greater than chance detection accuracy, thereby providing some initial evidence that MW detection is feasible in this context.

Aside from [25], other MW detection efforts have been limited to self-paced reading. In one of the first MW detection studies [10], students read aloud and then paraphrased biology paragraphs. They were periodically asked to report zone outs during reading on a 1 (all the time) to 7 (not at all) scale. Supervised machine learning models trained on acoustic-prosodic features to classify between "high" (1-3 on the scale) versus "low" zone outs (5-7 on the scale) achieved a 64% accuracy. However, this study did not adopt a student-independent validation approach, so it is unclear how well their detector would generalize to new students.

Other research has utilized log-file information to detect MW during self-paced reading. In one study [23], MW reports were collected via pseudo-random thought probes during self-paced computerized reading. Students responded either "yes" or "no" about whether they were MW at the time of the probe. Using textual features and reading behaviors from log-files, supervised machine learning models were able to detect MW with a 21% above-chance accuracy. Similarly, [12] attempted to predict MW during reading using textual features (e.g., difficulty, familiarity, and reading time), but it is not clear if their method, which utilized researcher-pre-defined thresholds, would generalize more broadly.

[a] Denotes equal contribution by authors.

Researchers have also adopted sensor-based approaches for MW detection during reading. Blanchard et al. [4] used an Affectiva Q sensor to record both galvanic skin response and skin temperature while participants read texts on research methods and periodically provided MW reports in response to thought probes. Their models attained a kappa value of .22 using a combination of peripheral physiology and contextual features (e.g., page numbers).

Eye gaze is perhaps one of the most promising modalities for MW detection due to the so called eye-mind link [27], which posits a coupling between eye movements and attentional focus. Several studies have thus built MW detectors using eye gaze features. The first study collected data from 84 students during self-paced reading of four texts on research methods [7]. MW reports were collected in response to thought probes triggered when gaze was fixated on predefined words on the screen. Supervised classification models were built from 27 gaze features and validated in a student-independent fashion. The authors achieved an accuracy of 60% after downsampling the data. Since downsampling was applied to both the training and test sets, it is unclear how the models would perform when presented with data that reflected the original skewed class distributions.

Their work was extended using a larger dataset of 178 students from two different universities and a wider array of 80 features, including blink and pupil features [2]. Students also read four texts on research methods, and MW reports were collected in response to nine pseudorandom probes that occurred between four to twelve seconds from the beginning of a page of text. Supervised models were built using an extended feature set and were cross-validated in a student-independent fashion. The models achieved an accuracy of 72% (31% above chance) when validated with a test set that maintained the original class distributions. Further, in [2], the authors provided evidence for the predictive validity of the model by showing that it predicted posttest scores at rates higher than self-reported MW, even after controlling for prior knowledge.

The results from this study indicate that MW can be detected from eye gaze during self-paced reading with moderate accuracy. However, there is an open question about the use of eye gaze to detect MW in additional contexts– in particular, for more complex stimuli like dynamic visual scenes. One study [35] provided evidence that eye movements can be predictive of attention while viewing short video clips. In this study, participants watched video clips in two different conditions: (1) without any distractions (attending) and (2) while performing a mental calculation (not attending). Results indicated that eye movements toward pre-determined salient locations in the scene could identify the watching condition (attending vs. not attending) with a 80.6% accuracy, albeit this is not quite the same as MW detection.

We should note that there is still some debate whether eye movements can be driven by salient features of the stimulus (*exogenous* control) or through conscious control (referred to as *endogenous* control). There is some research to suggest that eye movements are primarily driven by exogenous control. For example, previous research has shown that different viewers tend to fixate on the same locations [24], a phenomenon known as *attentional synchronicity,* which suggests exogenous control. However, other research pointed out that interesting objects are often the most visually salient [11]. Thus, it is possible that viewers fixate on the same locations because of top-down processes (endogenous control), as opposed to simply looking at what is salient. Additional evidence for endogenous control comes from a study which found that task instructions can have an effect on eye movements while viewing dynamic visual scenes [32]. The researchers found that participants looked at more peripheral and less visually salient areas of the scene when instructed in order to determine where the visual scenes were derived from compared to a general viewing task. Thus, eye movements related to endogenous control might be particularly revealing about MW. The current study utilizes this idea to compute features that capitalize on the relationship between eye movements and visually salient regions in the film.

1.2 Current Study and Novelty

In this paper we present one of the first attempts to automatically detect MW during narrative film viewing in a manner that generalizes to new students. We leverage what has been learned in previous work using eye gaze to detect MW during reading, while also developing theoretically-grounded features to improve detection accuracy in this novel context.

MW detection during film viewing poses unique challenges compared to reading, which has been the most common context for MW detection thus far. For one, eye movements are much more predictable during reading since the words on the screen are static. In addition, reading consists of fixations (periods where the gaze position is relatively stable) and saccades (rapid movements between fixations), while the dynamic nature of film also yields smooth pursuits (eye movements that follow a moving stimulus).

Second, the film played continuously without any clear breaks, presenting an additional challenge for MW detection. This is in contrast to reading tasks, which are segmented by page breaks. Thus, a novel method was devised to segment eye gaze data into instances for classification.

Finally, the dynamic nature of film allowed for novel content-dependent features that can be computed from dynamic areas of interest (AOI). Unlike reading, AOIs are particularly meaningful in a film viewing context because of the distinctive visual content areas that dynamically change throughout a film. In this study, AOIs were computed from both plot-related and visually salient regions.

2. DATA COLLECTION

This study utilized a subset of data reported by Kopp el. [21].

2.1 Participants

Eye gaze data was collected for 60 undergraduate students from a private Midwestern university. Students were 20.1 years old on average and 66% of the students were female.

2.2 Materials

Students watched "The Red Balloon," a 32.5 minute French film with few English subtitles (9 in all). The film was displayed on a computer screen with a resolution of 1920 × 1080. The film depicts the story of a young boy and a red balloon that follows him and can inexplicably move on its own. This film was chosen because it is unlikely that many students had previously seen it, which could have affected their propensity to mind wander. The film has also been used in previous film comprehension studies [36].

All data were collected using a Tobii TX 300 eye tracker that was attached to the bottom of the monitor. Eye gaze was recorded with a sampling frequency of 120 Hz for the first 14 participants (due to experimenter error), after which the sampling frequency was adjusted to 300 Hz. This difference was taken into account when filtering the gaze data as discussed below.

2.3 Mind Wandering Reports

Students were asked to self-report MW while they watched the film by pressing labeled keys on a standard keyboard. A short beep sounded to register their response, but the film was not otherwise interrupted. A self-caught MW report method was chosen as opposed to a probe-caught report method (where students are probed to report MW at pseudo-random intervals) in order to minimize disruption, which was critical as the film played without interruption.

Students were asked to differentiate between two different types of MW using separate keys: either task-unrelated thoughts (thoughts completely unrelated to the film such as upcoming vacation plans) or task-related interferences (thoughts related to the task but not the content of the film, such as "*This film is boring"*). For the present analyses, both task-unrelated thoughts and task-related interference were grouped as MW. There was a total of 616 MW reports. On average, students reported 10.3 instances of MW during the film (*SD* = 7.9l; *Min* = 1; *Max* = 31).

2.4 Procedure

Students were asked to sit comfortably at a desk in front of the monitor before beginning the eye-tracker calibration process. There were no restrictions on head movements, making the film viewing experience more ecologically valid than if a headrest was used. Students were randomly assigned to one of two conditions before the film started: in one condition, they read a short story explaining the movie plot [22] while students in the second condition read an unrelated baseball-themed story [1]. The experimental manipulations were part of a larger study and are not used here (more details can be found in [21]). Finally, students were given instructions for how to report MW and then the film began. Students completed a multiple choice comprehension assessment after viewing the film, but this data is not analyzed here.

3. MODEL BUILDING

3.1 Eye Movement Detection

Eye gaze was converted to eye movements (fixations, saccades, smooth pursuits, etc.) in order to filter out some of the inherent noise in raw eye gaze data. We first averaged the raw data from the right and left eyes. A simple moving average filter was then applied to the gaze points in order to smooth the signal while retaining the same sampling frequency. The filter used a window size of five samples for the 120 Hz data and seven samples for the 300 Hz data.

Eye movements were detected using a velocity based algorithm [18, 20]. These algorithms generally use thresholds to classify gaze points as fixations, saccades, or smooth pursuits. The algorithm first classified gaze points with a velocity greater than 110 degrees of visual angle/s as saccades. It then classified gaze points with a velocity lower than five degrees of visual angle/s as fixations. Any remaining gaze points were classified as smooth pursuits. The visual angle thresholds used were based on previous research [17].

3.2 Film Segmentation

Next, we segmented the continuous stream of eye gaze data into MW and non-MW segments. Each segment had three components: gap, window, and offset (see Figure 1). The *gap* was the number of seconds between adjacent segments and could be adjusted to change the ratio of MW to non-MW segments. The *window* was the portion of the segment used to compute features. The *offset* was the number of seconds between the MW report (the moment when the student pressed the key on the keyboard) and the end of the window. An offset was used in order to discard data affected by the student's motion to press the key when reporting MW. An offset size of three seconds was deemed appropriate based on observation of recorded videos.

The process began by creating a MW segment prior to each MW report (segment 2 in Figure 1). The data prior to the MW segment were then considered to be non-MW segments (segment 1) after accounting for the gap. There was no offset for non-MW segments as no key presses were involved.

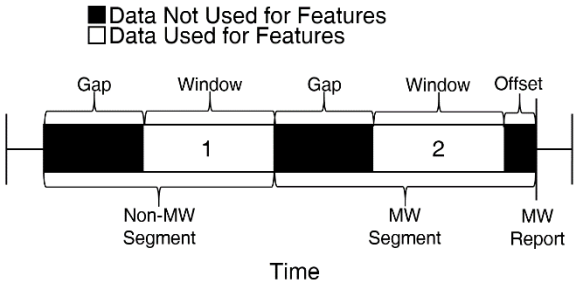


Figure 1. **Hypothetical example of segmented data**

There were several considerations when choosing the window and gap sizes. The segment size (sum of the window, offset, and gap sizes) determined both the number of available instances (segments) and the MW rate as shown in Table 1. Models were built with segment sizes of 45, 55, and 65 seconds, resulting in MW rates that ranged from .256 to .323 and number of instances from 2401 to 1626, thereby allowing us to explore how these two factors affected classification accuracy. For each of these segment sizes, the window size was also varied. In all, we considered window sizes of 10, 15, 20, and 25 seconds.

Table 1. Effect of segment size on number of segments and MW rate

Seg. Size (secs)	Number of Segs.	MW Rate
45	2401	.256
55	1931	.297
65	1626	.323

3.3 Feature Engineering

A total of 143 features were computed from the window in each segment. We considered global features, which were independent of the film content, and local features, which were content specific.

3.3.1 Global Features

There were 88 total global features. Of these, 75 were computed from measures of the eye movements, including fixations, saccades, and smooth pursuits, as well as blinks and pupil diameter. Fixation features were computed from the fixation *durations* (ms). Saccade features were computed from the *saccade durations* (ms), *amplitudes* (degrees of visual angle), *velocities* (degrees of visual angle/s), *relative angle* (degrees of visual angle between two consecutive saccades), and *absolute angle* (degrees of visual angle between a saccade and the x-axis). Smooth pursuit features were computed from the *duration* (ms), *length* (degrees of visual angle), and *velocity* (degrees of visual angle/s) of smooth pursuits. The following descriptive statistics of the distributions were used as the features: minimum, maximum, mean, median,

standard deviation, skew, kurtosis, and range. Counts of each eye movement type were also included as features.

Eight global features were obtained from pupil diameters, which were first z-score standardized at the student-level. The minimum, maximum, median, standard deviation, skew, kurtosis, and range were computed for the standardized pupil diameter distributions from each window and used as features.

There were five additional global features: blink count, mean blink duration, the ratio of total fixation duration to total saccade duration, the proportion of horizontal saccades, and the fixation dispersion.

3.3.2 Local Features

We identified two types of areas of interest (AOIs), Red Balloon AOIs and Visual Saliency AOIs, and computed features based on the locations of the AOIs in each frame. Red Balloon AOIs were used because the red balloon is one of the main objects in the film and endogenous attentional control might direct students to focus on these AOIs despite competing content. OpenCV [4], an open source computer vision software library, was used to isolate the red balloon from the rest of the image using a red color mask. A bounding box was drawn around a contour of the resultant image for each frame in which the balloon appeared (as shown on the left in Figure 2). Local features related to the red balloon were only computed for frames where it was present (58.2% of frames).

We manually examined each frame to ensure that the AOIs were computed correctly. The red balloon was present in 27,262 out of the 46,851 frames. An AOI was constructed for 26,925 of those frames, yielding an accuracy of 98.7%. The frames where the red balloon was missed could be attributed to lighting conditions (making the red balloon appear darker and thus difficult to distinguish from other parts of the scene), the small size of the red balloon, or the majority of the red balloon being off screen or occluded. These frames were left untouched. An additional 8 frames incorrectly had an AOI around an object that was not the red balloon. The AOI was simply deleted from these frames.

Visual Saliency AOIs were used because visually salient areas are known to attract eye gaze [11]. Although, the visual saliency and red balloon AOIs overlap in some cases, as in Figure 2, the visual saliency AOI can be computed for frames without the red balloon. The MATLAB implementation of a Graph-Based Visual Saliency algorithm [16] was used to produce a visual saliency map for each frame based on color, intensity, orientation, contrast, and movement. An area of no more than 2,000 pixels (1.1% of the screen area) surrounding the most salient point were retained and the remaining pixels were set to an intensity of 0. Similar to above, a bounding box was drawn around the largest contour of the processed image.

Local features were computed based on the relationship between the AOIs and each type of eye movement. The features included: (1) *AOI distance*, (2) *AOI intersection*, and (3) *saccade landing*. There were 32 AOI distance features, which captured the distance between the AOI and gaze positions. AOI distance features were computed as the distance between each fixation point or smooth pursuit point and the center of the AOI for each frame in the window. Fixation points were generated for each frame at the centroid of the fixation. Smooth pursuit points were generated for each frame using linear interpolation from the onset to the offset of each smooth pursuit. The minimum, maximum, mean, median, standard deviation, skew, kurtosis, and range of the measured distances were then computed for each eye movement, resulting in 16 features for each type of AOI (32 in all).

There were 12 additional AOI intersection features. These were calculated as the proportion of frames in which a fixation or smooth pursuit point was within the AOI bounding box. Four of these features used the original dimensions of the AOI bounding box. An additional eight used a bounding box expanded by either one or two degrees of visual angle in order to account for inaccurate eye gaze or cases where the AOI was small in size.

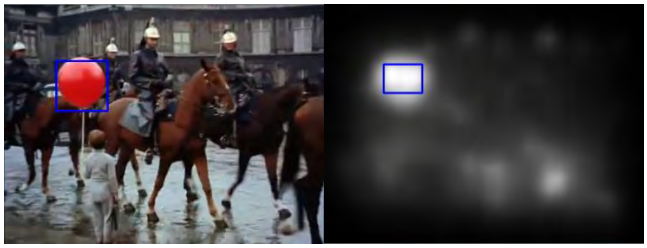


Figure 2. An example frame with a bounding box around contours of the red balloon (left) and most visually salient region (right)

Finally, there were 12 saccade landing features. For each AOI, there was a single feature that captured the number of saccades onto, away from, or within the AOI bounding box, which resulted in six features (3 per AOI). An additional six features were computed using a bounding box expanded by one degree of the visual angle to accommodate gaze tracking errors or small AOIs.

In all, there were 56 local features (32 AOI distance, 12 AOI intersection, and 12 saccade landing).

3.4 Model Building

Twelve supervised machine learning algorithms from Weka [14] were used to build models that discriminated between MW and non-MW instances (windows). The following classifiers were used: Bayes network; naïve Bayes; logistic regression; SVM; *k*-nearest neighbors; decision table; JRip; C4.5 decision tree; random forest; random tree; REPTree; and REPTree with bagging.

We also varied four external parameters: (1) feature type; (2) window and segment size; (3) feature selection percentage; and (4) sampling method. With respect to feature type, models were built with global features, local features, or both global and local features using feature-level fusion.

The segment and window size(s) were varied because there are various tradeoffs at play. Specifically, a larger segment size resulted in fewer instances but a higher MW rate, thereby reducing class imbalance. A larger window size afforded more data for each instance, but it also reduced the number of instances available for segments with the same gap size (e.g., a window size of 30 and gap size of 15 resulted in fewer instances than a window size of 40 and gap size of 15). Thus, models were built with segment sizes of 45, 55, or 65 seconds, and window sizes of either 10, 15, 20, or 25 seconds.

Feature selection was used on the training set of each cross-validation fold (see below). Features were ranked using correlation-based feature selection (CFS) [15] from Weka and the top 30%, 50%, or 80% of features ranked were retained.

Class imbalance poses a well-known challenge for supervised classifiers. Hence, *training* sets were resampled using

downsampling or oversampling. Downsampling consisted of randomly removing instances from the majority class (non-MW) until the two classes were balanced. Oversampling consisted of using the Synthetic Minority Over-sampling Technique (SMOTE) algorithm [5]. We also built models without any resampling for comparison purposes.

Table 3. Confusion matrices for best models

Feature Type	Actual	Classified		Prior
		Yes	*No*	
Global	*Yes*	.65 (hit)	.35 (miss)	.25
	No	.55 (FA)	.45 (CR)	.75
Local	*Yes*	.67 (hit)	.33 (miss)	.26
	No	.47 (FA)	.53 (CR)	.74
Global + Local	*Yes*	.68 (hit)	.32 (miss)	.25
	No	.60 (FA)	.40 (CR)	.75

Note: Values are proportionalized by class label
FA = false alarm; CR = correct rejection

Tolerance analysis was performed to address multicollinearity prior to building each model [9]. This consisted of removing features with a tolerance below .2, which indicates highly collinear features (such as number of fixations and number of saccades).

3.5 Model Validation and Evaluation

The models were evaluated using leave-one-student-out cross-validation, which ensures that data from each student is exclusive to either the testing set or training set. Feature selection and resampling were performed on the training set only. Feature selection was performed with data from a random 66% of students in the training data in each fold. Feature rankings were summed over five different random selections. Resampling was also repeated for five iterations in each training fold.

Models were evaluated using the F_1 score for the target class (MW), which was compared to the MW F_1 score of a chance classifier. For example, if the actual model classified 52% of the instances as MW, the chance classifier would classify a random 52% of the instances as MW. This resulted in a chance precision equal to the actual base rate of MW and a chance recall equal to the predicted MW rate. We believe this chance model to offer a more stringent comparison than a simple minority baseline (assign MW to all instances).

4. RESULTS

4.1 MW Detection Accuracy

The overall best performing model achieved a MW F_1 score of .45, compared to a chance MW F_1 score of .35, which is consistent with a 29% improvement above chance (Table 2). The model was a decision table classifier that used local features and had a window size of 20 seconds, segment size of 65 seconds, 11 features, and a downsampled training set. The confusion matrix for the model (Table 3) shows that the model makes fewer misses than false alarms.

Table 2. Performance metrics (F_1) for best models

Feature	F_1MW (Chance)	F_1 MW	F_1 Non MW	F_1 Overall
Global	.35	.39	.57	.53
Local	**.35**	**.45**	**.64**	**.59**
Global+ Local	.36	.39	.54	.50

The best global and global + local models were SVMs with a window size of 15 seconds, a segment size of 65 seconds, and a downsampled training set. The global model contained 5 features, while the global + local model contained 11 features. Both models achieved a lower MW F_1 score than the local feature model, due to much higher false alarm rates (see Table 3 and Figure 3)

With respect to the external parameters, no clear trends were observed for window size, segment size, or proportion of features selected, but downsampling and SMOTEing the training set outperformed no resampling method.

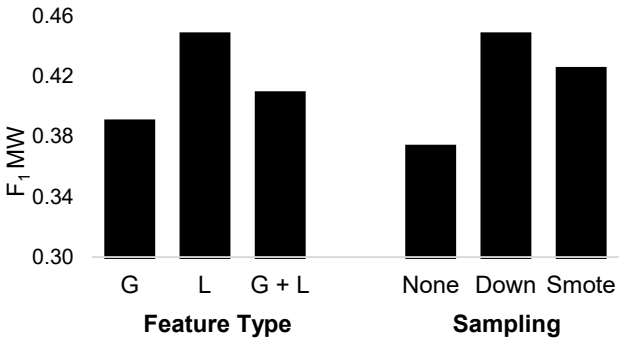


Figure 3. MW F_1 score for the best model by feature Type and resampling method. G = Global, L = Local, G + L = Global + Local; Down = Downsampling

4.2 Feature Analysis

We compared the mean values of each feature (computed per participant) for MW vs. non-MW instances with a two-tailed paired-samples *t*-test. We focused on the 16 global and 21 local features that were included in the best local and global models. Table 4 shows the effect size (Cohen's *d* – with positive values of *d* denoting higher values for MW compared to non-MW instances) for the significantly different ($p < .05$) features. We did not perform adjustments for multiple comparisons as the present analysis is exploratory in nature. Further, the number of significant findings (18%) is far greater than what we could achieve if we were capitalizing on chance alone.

We note that students were less likely to focus on the AOIs when they were MW. This is evidenced by a fewer number of frames where the smooth pursuit points intersected with the red balloon AOI or the most visually salient AOI. Further, there were fewer saccades onto and off of the most visually salient region during MW. Third, smooth pursuits had a longer range, but less variability in velocity during MW. Finally, there were fewer saccades during MW, which is consistent with previous findings of eye movements during MW while reading [2, 28]. Taken together, these results reflect a decoupling between salient regions

on the screen and eye movements, essentially signaling a breakdown in attentional synchronicity during MW.

Table 4. Effect size of difference in feature value between MW and non-MW instances

Feature	Cohen's *d*
Smooth Pursuit with Balloon AOI (frames)	-.37
Smooth Pursuit within 2° Saliency AOI (frames)	-.38
Number of Saccades away from Saliency AOI	-.39
Number of Saccades nearly onto Saliency AOI	-.35
Smooth Pursuit Duration Range (ms)	.30
Smooth Pursuit Velocity SD (°/s)	-.28
Number of Saccades	-.31

Note: SD = Standard Deviation; All tests were significant at $p < .05$ $df = 53$ for local features and $df = 50$ for global features.

5. DISCUSSION

There is a growing interest in assuaging the negative effects of MW during learning [6, 8]. Reliable MW detection is likely required to realize this goal. Although efforts in MW detection have had some success in the context of reading, MW detection in more media-rich contexts has been unexplored. As a step in this direction, this paper presents a student-independent detector of MW during narrative film comprehension, a context which is both timely and relevant given the increasing use of film and video lectures as educational resources.

5.1 Key Findings and Contributions

Our primary contribution is the computation of novel local gaze features that are based on the dynamic visual content of the film. Using these features, we were able to detect MW with a F_1 of .45 reflecting, a 29% improvement over chance. Furthermore, models built with local features outperformed models built with global features, or a combination of both global and local features. This suggests that taking the dynamic visual content into account (local features) can be more effective than merely tracking overall gaze patterns (global features), which has been the common method for MW detection during reading.

The local features likely performed better in the present context (narrative film viewing) compared to reading, because the unfolding visual stream provides cues as to where attention should be directed. Reading, in contrast, does not provide such explicit cues, so there is likely more variability in gaze patterns. This would explain why the global gaze features outperformed the local features during reading.

We also found that local features outperformed a combined local + global model, but we adopted a rather simplistic feature-level fusion strategy. It is an open question as to whether performance of the combined model could be boosted with more advanced fusion strategies.

Our results also provide insight into eye movements related to MW during film viewing. The key finding was that eye movements during MW were decoupled from the visually salient and important (balloon AOI) components of the visual stream, suggesting a breakdown in attentional control.

5.2 Applications

MW impedes comprehension by diverting a student's attention from the task at hand toward task-unrelated thoughts. Educational activities that involve comprehension from dynamic visual scenes, such as video clips or short instructional lectures, could benefit from pairing a MW detector with interventions that direct attention toward the learning task.

Beyond educational interfaces, detectors built from dynamic visual scenes have applications in entertainment and safety contexts. For example, they could be used to determine when viewers are more likely to MW while viewing entertainment films. The scenes could then be improved to increase viewer engagement.

Attentional focus is especially important for safety-critical tasks that require vigilance, such as air traffic control. MW detectors built for dynamic visual scenes might be more suitable for these types of tasks. However, empirical evidence is needed to determine the extent to which models built from narrative film viewing would generalize to these other contexts.

5.3 Limitations and Future Work

There were also some limitations with this study. The first limitation is the detection accuracy, which is moderate at best. It would be fruitful to explore improvements to the detector. Some possibilities include considering additional features based on other aspects of the visual content, such as faces or attempting more sophisticated modeling approaches that capture the unfolding temporal dynamics in eye gaze.

The segmentation method used in the study reflects yet another limitation as it rather arbitrarily segments the visual stream based on temporal windows. It would be worthwhile to explore content-based segmentation, such as scene transitions and event boundaries. This would also ensure consistent segments across students in lieu of the current method, which segments the film at different locations depending on the MW reports.

It is also unclear if the detector would generalize beyond the current film. "The Red Balloon" is a commercially produced film that employs cinematic devices to draw attention to the viewer [3]. In contrast, many instructional videos consist of an instructor lecturing to students [13] or lecturing over power point, which reflect rather different visual content.

Another limitation is the cost of eye tracking technology. The eye tracker used for this study was a cost-prohibitive Tobii TX300 that will not scale out of the laboratory. Fortunately, cost-effective eye tracking alternatives are becoming available, such as the Eye Tribe and Tobii EyeX, so replication with these trackers is warranted.

Finally, other limitations include a limited student sample (i.e. undergraduates from a private Midwestern college) and a laboratory setup. It is possible that the detector would not generalize to a more diverse student population or in more ecological environments. Retraining our model with data from more diverse populations and environments would be a suitable next step to increase its ecological validity.

5.4 Conclusion

We built the first student-independent gaze-based MW detector in the context of film viewing. The detector could be used to trigger interventions aimed at counteracting the negative effects of MW for an array of tasks involving dynamic visual scenes (e.g., watching instructional films, historic documentaries, or video lectures). Taken together, this work takes us closer to the goal of developing next-generation intelligent educational interfaces that "attend to attention" [6].

6. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation (NSF) (DRL 1235958 and IIS 1523091). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

7. REFERENCES

[1] Bernie the Early Bloomer - Bedtime Bedtime: *http://www.bedtime.com/bernie-the-early-bloomer/*. Accessed: 2016-02-09.

[2] Bixler, R. and D'Mello, S. 2015. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction*. 26, 1 (Sep. 2015), 33-68.

[3] Bordwell, D. 2013. *Narration in the fiction film*. Routledge, New York, NY.

[4] Bradski, G. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. 25, 11 (2000), 120-126.

[5] Chawla, N.V. et al. 2002. SMOTE: Synthetic Minority Over-Sampling Technique. *Journal of Artificial Intelligence Research*. 16, 1 (Jun. 2002), 321–357.

[6] D'Mello, S. et al. 2016. Attending to Attention: Detecting and Combating Mind Wandering during Computerized Reading. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, CA, May 07 - 12, 2016). *CHI EA '16*. ACM, New York, NY, 1661-1669.

[7] D'Mello, S. et al. 2013. Automatic Gaze-Based Detection of Mind Wandering during Reading. *Proceedings of the 7th International Conference on Educational Data Mining*. (Memphis, TN, Jul. 06 - 09, 2013) *EDM* '13. IEDMS, 364-365.

[8] D'Mello, S.K. 2016. Giving Eyesight to the Blind: Towards Attention-Aware AIED. *International Journal of Artificial Intelligence in Education*. 26, 2 (Jun. 2016), 645–659.

[9] Domingos, P. 2012. A few useful things to know about machine learning. *Communications of the ACM*. 55, 10 (Oct. 2012), 78–87.

[10] Drummond, J. and Litman, D. 2010. In the zone: Towards detecting student zoning out using supervised machine learning. In *Proceedings of the 10th International Conference on Intelligent Tutoring Systems* (Pittsburgh, PA, Jun. 14 - 18, 2010). *ITS '10*. Springer Berlin Heidelberg, 306–308.

[11] Elazary, L. and Itti, L. 2008. Interesting objects are visually salient. *Journal of Vision*. 8, 3 (Mar. 2008), 3–3.

[12] Franklin, M.S. et al. 2011. Catching the mind in flight: using behavioral indices to detect mindless reading in real time. *Psychon Bull Rev*. 18, 5 (Oct. 2011), 992–997.

[13] Guo, P.J. et al. 2014. How video production affects student engagement: an empirical study of MOOC videos. *Proceedings of the first ACM conference on Learning@ scale conference* (Atlanta, GA, Mar. 04 - 05, 2014), ACM, 41–50.

[14] Hall, M. et al. 2009. The WEKA Data Mining Software: an Update. *ACM SIGKDD Explorations Newsletter*. 11, 1 (2009), 10–18.

[15] Hall, M.A. 1999. *Correlation-Based Feature Selection for Machine Learning*. Doctoral Thesis. Department of Computer Science, The University of Waikato.

[16] Harel, J. et al. 2006. Graph-based visual saliency. In *Proceedings of the Advances in Neural Information Processing Systems* (Vancouver, BC, Dec. 04 - 06, 2006), *NIPS* '06. MIT Press, Cambridge, MA, 545–552.

[17] Holmqvist, K. et al. 2011. *Eye Tracking: A Comprehensive Guide to Methods and Measures*. Oxford University Press, Oxford, UK.

[18] Karpov, A.V. and Komogortsev, O. 2013. Automated Classification and Scoring of Smooth Pursuit Eye Movements in Presence of Fixations and Saccades. *Journal of Behavioral Research Methods.* 45,1 (Mar. 2013) 203-215.

[19] Killingsworth, M.A. and Gilbert, D.T. 2010. A Wandering Mind is an Unhappy Mind. *Science*. 330, 6006 (Nov. 2010), 932–932.

[20] Komogortsev, O.V. et al. 2010. Standardization of automated analyses of oculomotor fixation and saccadic behaviors. *Biomedical Engineering, IEEE Transactions on*. 57, 11 (Jul. 2010), 2635–2645.

[21] Kopp, K. et al. 2015. Mind wandering during film comprehension: The role of prior knowledge and situational interest. *Psychonomic Bulletin & Review*. (Sep. 2015), 1-7.

[22] Lamorisse, A. 1956. *The Red Balloon*. Penguin Random House LLC, New York, NY.

[23] Mills, C. and D'Mello, S. 2015. Toward a Real-time (Day) Dreamcatcher: Sensor-Free Detection of Mind Wandering During Online Reading. *Proceedings of the 8th International Conference on Educational Data Mining*. (Madrid, Spain, Jun. 26 - 29, 2015) *EDM* '15. IEDMS, 69-76.

[24] Mital, P.K. et al. 2011. Clustering of Gaze During Dynamic Scene Viewing is Predicted by Motion. *Cognitive Computation*. 3, 1 (Mar. 2011), 5–24.

[25] Pham, P. and Wang, J. 2015. AttentiveLearner: Improving Mobile MOOC Learning via Implicit Heart Rate Tracking. In *Proceedings of the 17th International Conference on Artificial Intelligence in Education*. (Madrid, Spain, Jun. 22 - 26, 2015). *AIED '15*. Springer International Publishing, 367–376.

[26] Randall, J.G. et al. 2014. Mind-wandering, cognition, and performance: A theory-driven meta-analysis of attention regulation. *Psychological Bulletin*. 140, 6 (Nov. 2014), 1411–1431.

[27] Rayner, K. 1998. Eye Movements in Reading and Information Processing: 20 Years of Research. *Psychological Bulletin*. 124, 3 (Nov. 1998), 372.

[28] Reichle, E.D. et al. 2010. Eye Movements During Mindless Reading. *Psychological Science*. 21, 9 (Aug. 2010), 1300–1310.

[29] Risko, E.F. et al. 2012. Everyday attention: variation in mind wandering and memory in a lecture. *Applied Cognitive Psychology*. 26, 2 (Apr. 2012), 234–242.

[30] Smallwood, J. et al. 2007. Counting the cost of an absent mind: Mind wandering as an underrecognized influence on educational performance. *Psychonomic Bulletin & Review*. 14, 2 (Apr. 2007), 230–236.

[31] Smallwood, J. and Schooler, J.W. 2006. The Restless Mind. *Psychological Bulletin*. 132, 6 (Nov. 2006), 946–958.

[32] Smith, T.J. and Mital, P.K. 2013. Attentional synchrony and the influence of viewing task on gaze behavior in static and dynamic scenes. *Journal of Vision*. 13, 8 (Jul. 2013), 16–16.

[33] Szpunar, K.K. et al. 2013. Mind wandering and education: from the classroom to online learning. *Frontiers in psychology*. 4 (Aug. 2013), 1-7.

[34] Tan, T. et al. 2015. Mind Wandering and the Incubation Effect in Insight Problem Solving. *Creativity Research Journal*. 27, 4 (Nov. 2015), 375–382.

[35] Yonetani, R. et al. 2012. Multi-mode saliency dynamics model for analyzing gaze and attention. *Proceedings of the Symposium on Eye Tracking Research and Applications* (Santa Barbara, CA, Mar. 28 - 30, 2012). ETRA ' 12, ACM, 115–122.

[36] Zacks, J.M. 2010. The brain's cutting-room floor: segmentation of narrative cinema. *Frontiers in Human Neuroscience*. 4, 168 (Oct. 2010), 1-15.

Riding an emotional roller-coaster: A multimodal study of young child's math problem solving activities *

Lujie Chen
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA USA
lujiec@andrew.cmu.edu

Xin Li
Shandong Yingcai University
No. 2 Yingcai Road
Shandong, China
lixin@ycxy.com

Zhuyun Xia
Shandong Yingcai University
No. 2 Yingcai Road
Shandong, China
xiazhujun@ycxy.com

Zhanmei Song
Shandong Yingcai University
No. 2 Yingcai Road
Shandong, China
songzhanmei@ycxy.com

Louis-Philippe Morency
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA USA
morency@cs.cmu.edu

Artur Dubrawski
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA USA
awd@cs.cmu.edu

ABSTRACT

Solving challenging math problems often invites a child to ride an "emotional roller-coaster" and experience a complex mixture of emotions including confusion, frustration, joy, and surprise. Early exposure to this type of "hard fun" may stimulate child's interest and curiosity of mathematics and nurture life long skills such as resilience and perseverance. However, without optimal support, it may also turn off child prematurely due to unresolved frustration. An ideal teacher is able to pick up child's subtle emotional signals in real time and respond optimally to offer cognitive and emotional support. In order to design an intelligent tutor specifically designed for this purpose, it is necessary to understand at fine-grained level the child's emotion experience and its interplay with the inter-personal communication dynamics between child and his/her teacher. In this study, we made such an attempt by analyzing a series of video recordings of problem solving sessions by a young student and his mom, the ideal teacher. We demonstrate a multimodal analysis framework to characterize several aspects of the child-mom interaction patterns within the emotional context at a granular level. We then build machine learning models to predict teacher's response using extracted multimodal features. In addition, we validate the performance of automatic detector of affect, intent-to-connect behavior, and voice activity, using annotated data, which provides evidence of the potential utility of the presented tools in scaling up analysis of this type to large number of subjects and in implementing tools to guide teachers towards optimal interactions in real time.

*(Does NOT produce the permission block, copyright information nor page numbering). For use with ACM_PROC_ARTICLE-SP.CLS. Supported by ACM.

Keywords

math problem solving, affect, interaction dynamics, multimodal learning analytics

1. INTRODUCTION

A popular perception of math education in the US schools is often associated with the lack of inspiration and excitement. One of the possible reasons for that is a common perception of math learning as shallow learning activities such as memorizing multiplication tables and procedure learning activities such as long division [10]. This is especially true with elementary level education where learning facts and procedures accounts for most of the curriculum. In contrast, math problem solving activities can take a form of complex learning [10] that often requires the student to take an adventurous emotional and cognitive "roller-coaster" ride when navigating the uncharted land of possible solutions.

Involvement in this type of activities from young age may play a major role in stimulating student's interest in math and more generally in STEM topics. It may also help building self-confidence and perseverance. However, if not done right, it may disengage student due to unresolved frustration and result in an even more negative view of the subject. It is thus important to know what is the right mixture of emotional and cognitive support to be provided in the process, as well as the right amount and the optimum timing of such support. This role of support is consistent with the vision of a Learning Companion [12] which is a computer system that facilitates learning on the side, is watchful for the trajectory and provides appropriate level of support.

In this study, we explore that question by analyzing the fine-grained multimodal behavior cues that could be automatically extracted from video recordings of one-to-one math problem solving sessions in a naturalist environment. Specifically, we explore data driven methods to characterize the temporal dynamics of the child's emotion states as well as patterns of the interaction between the child and the teacher when problem solving processes unfold.

2. RELATED WORK

A substantial amount of prior work on the automatic detection of student's affective states exists primarily in the context of intelligent tutor systems. [2] introduces a "sensor free" detector to infer engagement from the logs of students' interaction with computerized reading tutor using a method called engagement tracing. [15] uses facial expression analysis to infer engagement during interactive cognitive skill training sessions. Using the same sensing modality, [13] studies an array of affective states such as boredom, confusion, delight, flow, frustration and surprise, based on Facial Action Units. [5] leverages multimodal inputs including conversational cues with computer tutors and gross body language as well as facial features to detect distinct affective states.

While the work mentioned above focuses on static modeling of affects, another thread studies dynamics of affective states. [5] characterizes transitions of affective states between confusion, engagement/flow, boredom and frustration during complex learning activities when using computer tutors. [11] uses a hierarchical dynamic Bayesian network to model temporal dynamics of behavior trends such as flow, stuck and off-task, as well as related emotion states such as stress, confusion, boredom and frustration.

Within literature on student and human teacher interaction, [14] applied theory of dynamic systems to model real time teacher-student interactions using videotaped classroom sessions. Quality of interaction was rated and analyzed in terms of content, structure and complementary. [8] uses turn level audio features and contextual information to predict students' high level affect states using a human-human tutoring dialogue corpus.

There are several aspects in which this study differs from relevant prior work: (1) Instead of using computer tutor, we are interested in an "unplugged" scenario where the child is interacting with a real human teacher. This setup allows us to observe the genuine inter-personal communication dynamics which is not available when interacting with a computer tutor. Specifically, help seeking behaviors, a well studied phenomenon with computer tutors, are generalized into Intent-To-Connect (ITC) behaviors manifested by either subtle cues such as eye contacts or head pose changes, or explicit verbal help requests. ITC behaviors carry a richer meaning that exceeds the conventional cognitive support oriented "help seeking". Instead, ITC behaviors can also be used to signal emotional connection for other purposes such as "comfort seeking" or "joy sharing"; (2) The subject in this study is a child at young age. Since children at this age often are not exposed to the social pressure to hide negative emotions such as frustration, this allows observing their emotions with high fidelity, though it also presents unique detection challenges since the frequent baseline body movement are more frequently observed in young children; (3) The problem solving tasks in this study call for the child to take an active role in open exploration, with support from adult only when needed, whereas other studies typically consider a specific task such as cognitive skill training [15]. Consequently, we expect to observe non-baseline affect states at higher level of frequency and intensity; (4) With a few exceptions, most of the existing work relies on signals from a single modality, while this study attempts to integrate multimodal signals available from audio and video data.

```
At a round table there are chairs placed with the same
distance between them.  They are numbered consecutively
1, 2, 3, ?.  Peter is sitting on chair number 11, di-
rectly across from Chris, who is sitting on chair num-
ber 4.  How many chairs are there at the table?
A) 13     B) 14     C) 16     D) 17     E) 22
```

Figure 1: An example of a Math Kangaroo problem

3. DATASET AND USER STUDY

We collected video recordings of one-to-one problem solving sessions between a 9-year-old boy (a third grader) and his mom (the first author of this paper) as his teacher. We chose this setup because this mom and son has worked together on math problem solving for a few years. As result, the mom is used to picking up and reacting optimally to child's behaviors. This is the closest to the desirable model of the "ideal teacher" as we described earlier.

In each of multiple sessions, the child was asked to solve one challenging math problem. We selected the problems from Math Kangaroo[1], an annual international math competition for students in K-12. Using interesting but challenging problems, the goal of this competition is to stimulate students' interest in math problem solving. There are 24 problems in each competition, divided into three sections with gradual increase of difficulty. The problems for this study were selected from the most difficult set of levels 3 and 4 competition geared towards students in third and fourth grades. Those problems assume basic arithmetic skills and background knowledge at the child's grade level. Figure 1 shows an example of a problem used in the study. In all of the sessions, mom tried to optimize the experience of the child by balancing the goal of reducing frustration and providing sufficiently stimulating challenge.

The videos were captured in a home environment using a Logitech 1080P webcam with an integrated microphone. The positions of mom and child make it possible to capture child's non-verbal behavior cues such as head pose and gaze changes when he intends to connect with mom. Both audios and videos were captured for child, whereas only voice was recorded for mom. We recorded a total of 21 sessions, accumulating 141 minutes of raw video with mean length of 6.4 minutes per session, with longest session lasting 14.6 minutes and the shortest only about 2 minutes. In most of the recordings, the child ended with a joyful mood and a sense of accomplishment.

All recordings were manually annotated in ELAN [2][3] for voice activity at utterance level of child and mom. We also annotated child's non-verbal ITC behaviors using cues such as head turn and eye contact as well as verbal cues. Annotation included timestamps of start and end of events. Frame-

[1]www.mathkangaroo.org

[2]http://tla.mpi.nl/tools/tla-tools/elan/, Max Planck Institute for Psycholinguistics, The Language Archive, Nijmegen, The Netherlands

Table 1: The affective state and problem solving stages and their behavior cues F(facial), HP(head pose), Voc(vocal), Ver(verbal)

Affects	Problem Understanding	Planning	Execution
Confusion	F+Ver		
Frustration		F+HP+Voc	F+HP
Joy		F+Ver	
Engaged			Voc+HP
Disengaged	HP	HP	HP

by-frame emotion states were extracted using FACET Software Development Kit[3]. Head pose and gaze features were extracted using OpenFace framework toolkit[4] [1]. In addition, acoustic features were extracted using COVAREP toolkit (version 1.3.2) [4] every 10ms.

4. QUALITATIVE ANALYSIS

4.1 Problem solving stages and affective states

In his famous book "How to solve it" [9] , the mathematician Gorge Polya proposed four stages of problem solving, a framework widely used in today's math problem solving instructions. In this study we adapt it into a three-stage framework without the last reflection stage, including "problem understanding", "planning" and "execution". Table 1 lists the most likely affective states as well as plausible behavior cues at each problem solving stage based on qualitative analysis of video recordings. Those cues were used to guide the annotation of events as well as informed the feature design for the automated analysis.

There are several "landmark" behavior cues that could be used to identify problem solving stages and transitions. During problem understanding stage, the child reads the problem and asks clarification questions when necessary. The child often ends this stage by saying "okay". Afterwards, the child might be stuck at the planning stage with no idea as for how to proceed, or go on smoothly with a brief planning stage, or in rare cases dive right into the implementation stage. During the implementation stage, the child is often engaged, with his head down, writing on paper, either silently or with fast paced talking suggesting a "flow" experience. After one attempt, he may succeed at solving the problem, or he could find that his answer is obviously wrong.[5] In those cases, he needs to re-enter into the planning stage to find alternative solution, or rework the original plan. The process ends when the correct answer is confirmed in which case the child often exhibits positive emotions such as excitement and joy.

4.2 Interpersonal communication dynamics

The problem solving sessions can be highly interactive between mom and child: the child actively verbalizes his problem solving process and frequently connects with mom through verbal and non-verbal cues which we call "intent-to-connect" behaviors, or, ITC. Verbal ITC cues refer to explicit request for help or questions, while non-verbal ITCs are subtle cues of head pose and/or gaze change.

ITC may carry multiple different meanings, which calls for differentiated responses to achieve best learning outcomes. According to her interpretation, mom's response to ITC may serve a purely cognitive support purpose such as providing scaffolding, or, as in most cases, providing emotional support in the form of "back channel" signals such as "yes", "good", "good thinking". Given the many subtle variation of ITCs that can be considered in modeling response, it is desirable to take into account contextual information such as problem solving stages and emotion states in order to infer the true intent of an ITC.

Figure 2 provides an overview of the events of an example session that illustrates the interplay between interpersonal communication dynamics, including voice activity events (mom's talk and child's talk) and child's ITC behaviors, within the context of problem solving stages transitions and emotion states. As shown in the plot, the session started with the problem understanding stage (1) that is characterized by child's monologue while reading the problem followed by a brief period of pause and thinking. At the same time, confusion and frustration began to kick in (A), after which mom started to intervene by explaining the problem (2), then child entered planning and execution stage (3) that lasts about 3 minutes. Then, at 1 minute into this process, child said "I didn't get it" with head turn, and mom offered help by asking "Do you need help?". However, the child did not take the offer and kept working on his own. Towards the end of this phase, the child exhibited positive emotion of joy. Then mom discovered that child is on the wrong path, so she intervened (4) and the two worked together to correct the error during which time the child showed brief moments of frustration and confusion (C). Afterwards, the session moved into the problem solved stage (5), the child revealed a spike of surprise and moderate joy (D).

5. QUANTITATIVE ANALYSIS

In this section, we present an analytic framework developed to characterize and understand the interplay between dynamics of emotional states as well as interpersonal communication. We first present a method to quantify the relationship between ITC and mom and child's talk. We then present results from analysis of videos using emotion and interaction features. We end this section with predictive modeling of mom's response using multimodal features.

5.1 Interpersonal communication dynamics

5.1.1 Event intensity metric

We use event intensity metric to characterize temporal patterns of intensity of a specific type of event (e.g. child's talk). This metric takes into account both the frequency and duration of an event. To compute the metric, we first convert the annotated duration of the events into discrete sequences sampled uniformly at interval of every 20ms. Binary flag of 1 is assigned to intervals of the event's occurrence and 0 otherwise. A moving sum is then computed from a window centered at the time of interest. The resulting time series of the moving sum of thusly assigned binary flags characterizes

[3]www.emotient.com
[4]https://github.com/TadasBaltrusaitis/OpenFace
[5]Since the problems are formulated as multiple-choice questions, if the answer is not any of the choices provided, then it must be wrong

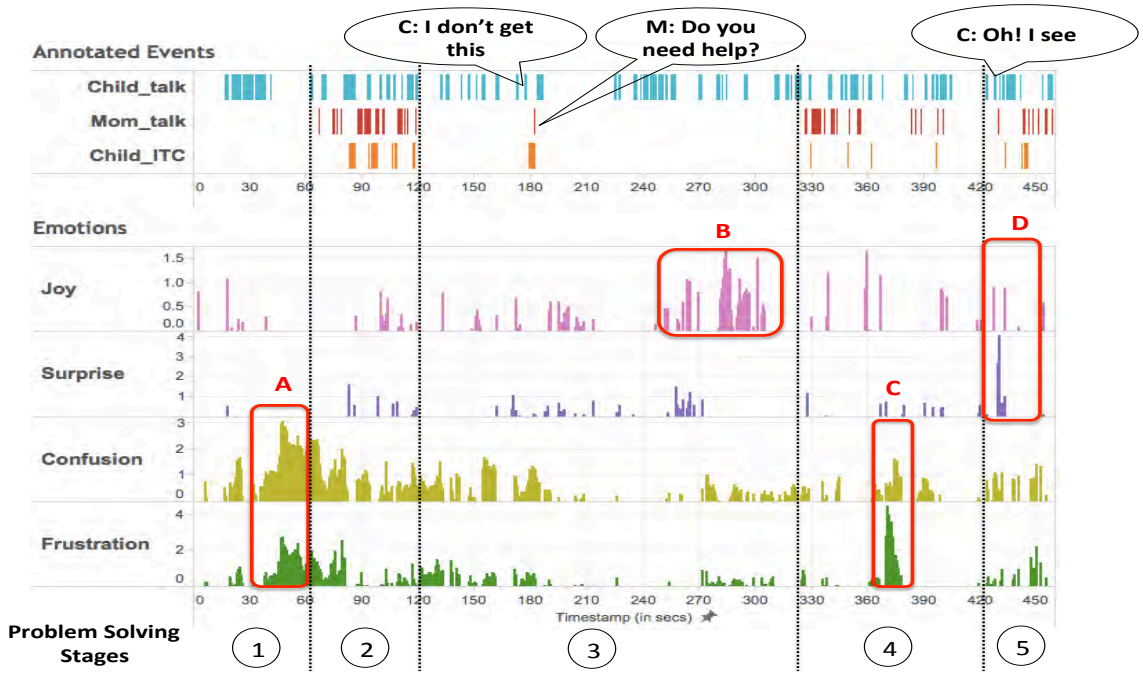


Figure 2: Timeline of annotated events within the context of problem solving stages and affective state transitions. Problem solving stages: (1) problem understanding (2) mom's intervention (3) planning and execution (4) mom's intervention (5) solved ; Emotion states: (A) confused and frustrated (B) joy (C) confused and frustrated (D) joy and surprise; Dialogue legends: C: child; M: mom

temporal intensity distribution of the events. The width of the window determines temporal resolution and smoothness of the temporal patterns.

5.1.2 *Floor sharing metrics*

We characterize temporal patterns of floor sharing between mom and child using normalized metrics of event intensity of mom's talk and child's talk as described above. The formula for mom's sharing of conversation at time stamp t is given as:

$$Mom\_Talk\_Share(t) = \frac{Mom\_Talk(t)}{Child\_Talk(t) + Mom\_Talk(t))} \quad (1)$$

This metric is useful to identify periods of time when mom's intervention dominates or vice versa. Figure 3 shows temporal distribution of floor sharing patterns for each video sorted by video length. It seems apparent that in short videos (presumably representing easy problems), mom did not talk much. However, longer videos often involve larger proportion of mom's talk. It is also interesting to observe that mom's talk often occurs in batches, presumably at the time when child gets stuck so that elaborate explanation is necessary.

5.1.3 *Synchronization of voice activity and ITC*

In this section, we describe a method to quantify synchronization between voice activity (mom's talk and child's talk) and ITC. Figure 4 shows two examples with different synchronization patterns. In the left plot, ITC seems to be more synchronized with child's talk, while in the right plot it is more synchronized with mom's talk which suggests child's attention or engagement . We summarize synchronization as the pairwise correlation among these time series. The result is displayed in the scatter plot in Figure 5 in which each video is plotted as a point labeled with its index. As shown, ITC seems to be more correlated with mom's talk than child's talk as seen from the cluster of points in the upper left quadrant of the plot in Figure 5, with a few exceptions (videos 12, 14 and 32) in which ITC seems to be drifted away from mom's talk and correlate more with child's talk. Incidentally, mom intervened significantly in those videos, which suggests child's disengagement may be induced by mom's higher intensity of teaching.

5.2 Video analysis

In this section, we report the results from video analysis by exploring the pairwise statistical correlations among variables related to interaction dynamics (i.e. voice activity and ITC behaviors) and affective states, as well as the outcome measure, i.e. time taken to solve a problem. For each video, we computed the following variables:

1. Interaction dynamics variables
   - Mom/Child talk ratio (mom-child): The ratio of the accumulative duration of mom's talk versus child's talk.
   - ITC rate: The count of ITC, normalized by video length.

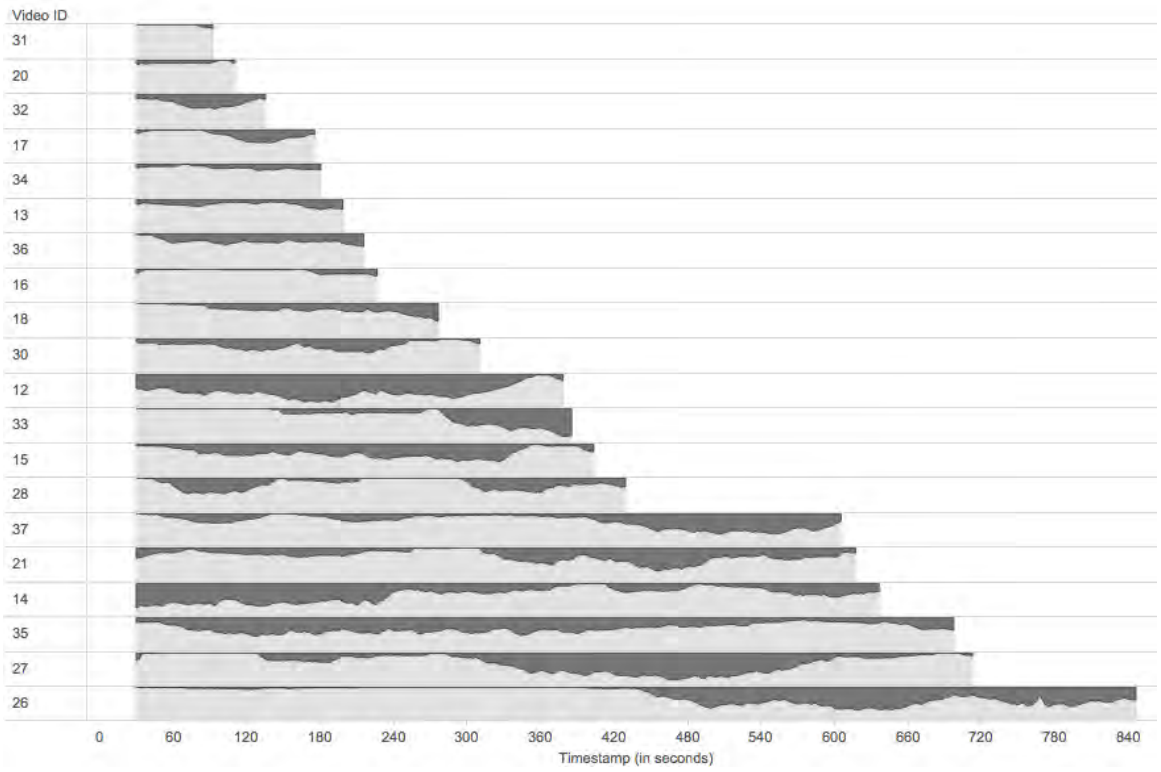


Figure 3: Temporal patterns of floor sharing for each video (dark color: mom, light color: child)

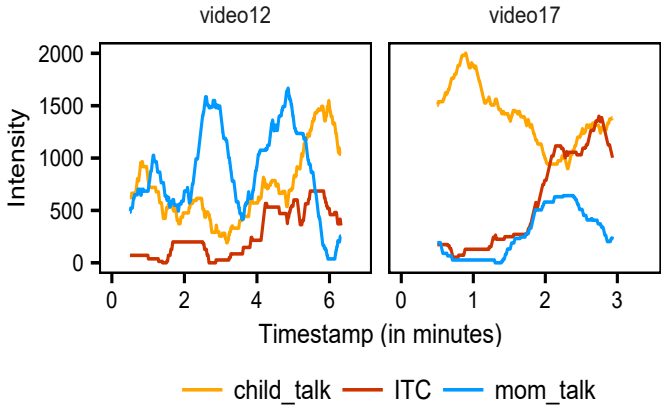


Figure 4: Two example time series plot of events intensity, ITC synchronized more with child's talk (left) or with mom's talk (right)

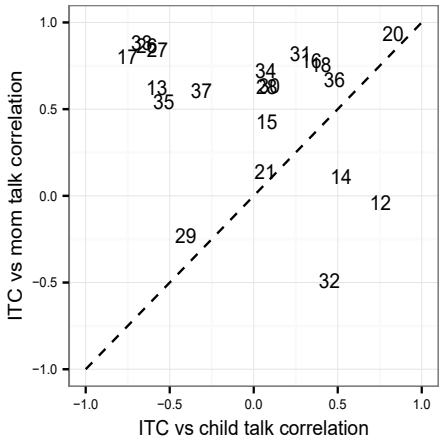


Figure 5: Summary of synchronization : ITC vs child talk (x axis) and ITC vs mom talk (y axis)

- Mom's back channel response rate (mom-BC): Back channel response is defined as a response that lasts less than 2 seconds. This variable represents the count of such response normalized by video length.

2. Affective state variables

   - These are counts of video frames with FACET score greater than 1, normalized by total number of frames during the period of interest for each of the four affect channels including joy, surprise, frustrations and confusion.

In order to further explore the importance of features at the beginning as well as those at the end of a session,we also compute statistical features from two sub-periods of interest: first 30 and last 30 seconds of each video.

We then compute pairwise Pearson correlation among the variables, including outcome. Due to the small number of videos, for each pair of correlations, we performed 1000 iterations of a randomization test [7] under null hypothesis of zero correlation to obtain non-parametric p-values. A sparse graph (Figure 6) is created to summarize the significant correlations among the variables with a p-value cutoff at 5% significance level.

There are several interesting insights that could be derived from this graph. Firstly, there is a significant positive correlation between initial frustration or confusion and the time taken to solve a problem. Since the beginning period is likely to be devoted to problem understanding, this suggests difficulty in understanding of the problem is the first obstacle child may face. His struggle in this period is likely to extend over the entire problem solving process. Secondly, there is a positive correlation between mom/child talk ratio and the video length. This suggests that mom intervenes more in case of hard problems which take longer to solve. Thirdly, child's ITC rate is positively correlated with mom's back channel rates which suggests a level of interaction synchrony between the two. Lastly, there is negative correlation between the overall frustration and joy at the ending period, in other words, more frustrating experience is associated with less joy toward the end, and vice versa.

5.3 Predictive modeling of response

In this section, we report the results from machine learning models used to predict the binary label if there is mom's response within 5 seconds for occurrence of an ITC. The following list explains the features used for the predictive model:

1. Voice activity features:

   - ITC co-occurrence: The count of other ITC within time windows of 2, 5 and 10 seconds respectively for each ITC;
   - Overlap statistics: The number of child talk, mom talk and child or mom talk events that are overlapping a given instance of ITC;

Table 2: Performance of the predictive models of mom's response to child's ITC (leave one video out)

Model	AUC mean	Lower bound of CI	Upper bound of CI
LR	0.594	0.557	0.630
NB	0.617	0.581	0.652
SVM	0.519	0.506	0.531

2. Head pose features : Min, max, mean, median of detection success, confidence, tilt, turn, up-down, within 5 seconds surrounding a given ITC;
3. Features from affect detector: Min, max, mean, median of FACET score for each of the emotion categories (joy, surprise, confusion, frustration and baseline) within the 5 seconds surrounding a given ITC. Negative scores are replaced with 0.

We performed a leave-one-video-out cross-validation experiment to evaluate three different classifiers (logistic regression[LR], naive bayes[NB] and support vector machine[SVM]). The Area Under Curve(AUC) score for each classifier is shown in Table 2 with mean values and 95% confidence intervals. Though the overall performance has much room for improvement, all of the three models perform significantly better than random, which suggests there are indeed predictive signals in the features. A better model might need to incorporate features related to the problem solving state, which may be learned using state space method such as Hidden Markov Models or Conditional Random Fields.

6. VALIDATION OF AUTOMATIC RECOGNITION

6.1 ITC and voice activity recognition

In this section we summarize the results from following recognition tasks:

1. ITC recognition using Openface head pose features. For each video, a random sample of 500 positive frames with ITC and 500 negative frames without ITC were selected, and a model was trained using frame-by-frame head pose features (confidence, Tx, Ty, Tz, Rx, Ry, Rz, up-down, turn and tilt) as inputs;
2. Voice activity recognition using features from COVARAP. One classifier built to discriminate between speaker and non-speaker segments; another classifier to discriminate mom's talk and child's talk. For each task, we random select 500 samples from each class from each video.

In those recognition tasks, we experimented with different types of classifiers including logistic regression, support vector machine, decision tree and naive Bayes, and found logistic regression to show overall superior performance as reported in Table 3. We performed leave-one-video-out cross validation and reported mean AUC scores. We also reported per video performance where we build a dedicated classifier for each video and summarized 10-fold cross-validation AUC score across all videos. As expected, leave-one-video performance is worse than the per video performance for both ITC

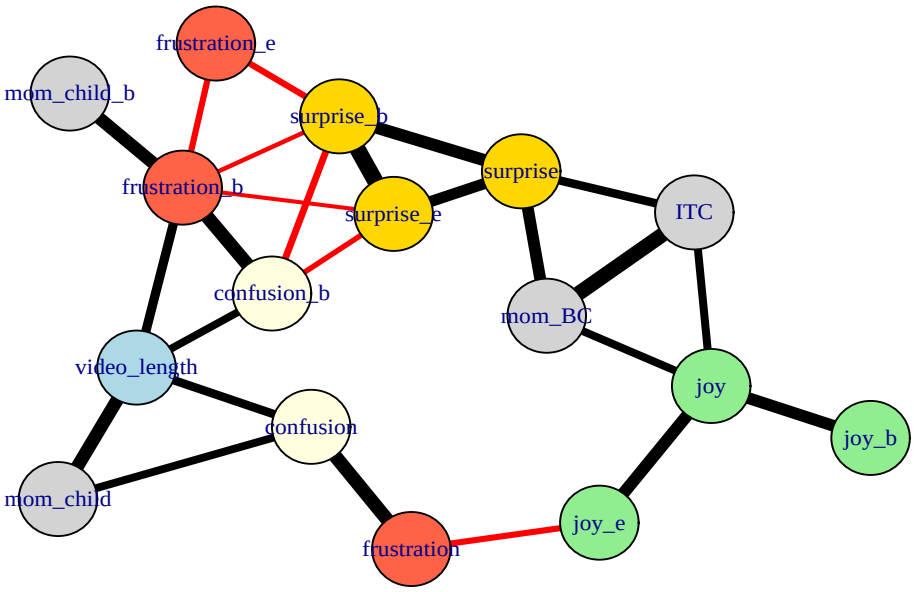


Figure 6: Graph of pairwise correlation of variables. Variable named in the form of "xxx" (e.g. joy) are computed from full length of video; "xxx_b" (e.g. mom_child_b) are computed from first 30 seconds of each video, "xxx_e" (e.g. surprise_e) are computed from the last 30 seconds. Black edges depict positive correlations while red edges represent negative correlations. The width of the edge corresponds to the magnitude of the absolute value of the correlation. The colors of the nodes denote types of variable: Green-Joy, Red-Frustration, Golden-Surprise, Light Yellow-Confusion, Gray-Interpersonal dynamics features, Blue-Outcome

Table 3: AUC scores of models built for ITC and voice activity recognition task

	ITC recognition	Speaker vs. non-speaker	Mom and child talk
Leave one video out CV	0.90	0.81	0.74
Dedicated classifier 10-fold CV	0.92	0.81	0.81

and mom and child talk classification. This suggests that camera and microphone calibration/normalization might have impact on those two tasks, however the speaker and non-speaker classification task seems to be more robust to this issue. Overall, performance of ITC detection is satisfactory, while the voice activity recognition task leaves room for improvement, using a higher quality microphone for each participant might be beneficial.

6.2 Affect detection

In this section, we report validation results for affect labels produced by FACET. We randomly selected 30 top-scored frames (at least 10 seconds apart) from each of the affect class (joy, surprise, frustration, confusion and baseline), and requested labels from two independent annotators who were blinded from FACET labels. Table 4 shows Cohen's Kappa for each affect label (when treated as a binary labeling task) as well as the overall score. As shown, the inter-rater agreement is relatively high for both joy and surprise, though the annotator's agreement with FACET is higher for joy

Table 4: Validation scores of FACET's affect detection (Cohen's Kappa)

Affect	annotator1 vs FACET	annotator2 vs FACET	annotator1 vs annotator2
joy	0.70	0.57	0.73
surprise	0.48	0.43	0.71
confusion	0.30	0.51	0.41
frustration	0.11	0.36	0.44
baseline	0.58	0.42	0.44
overall	0.35	0.46	0.41

than surprise. Confusion and frustration are two of the most challenging affects to detect as compared to joy and surprise, possibly due to the fact that confusion and frustration are easily mistaken for each other, as evidenced by the low inter-rater agreement score. This suboptimal performance may also be attributable to the fact that FACET is trained on faces from general population rather than specifically on young children. A detection algorithm that would incorporate transfer learning and age based customization will possibly improve the performance.

7. CONCLUSION AND FUTURE WORK

In this study, we analyzed data from the 21 video recordings of a nine year old boy while he was working through challenging math problems that demand high order cognitive skills to understand, plan, execute and solve the problems on his own, with only limited and mostly passive support from his mom.

We have shown qualitatively that there are clusters of non-baseline emotions rolling throughout the problem solving process, with the strongest representation from emotion class of joy, surprise, confusion and frustration. This observation confirmed our hypothesis that this type of active exploration indeed facilitates a unique experience of riding an "emotional roller coaster".

We also explored various analytical approaches to characterize the interpersonal dynamics between mom and child as well as the interplay with ITC behaviors. Our video analysis reveals some interesting associations between voice activity, ITC and emotional context.

Lastly, we built a classification model to predict whether there is mom's response within 5 seconds of a given ITC. The recognition task results show promise for automatic annotation of ITC and voice activity in order to scale up the presented analysis. Those findings collectively provide initial evidence for the feasibility of building affect sensitive computer tutor by mining multimodal signals as demonstrated in this study.

The key contributions of this paper include the new framework for fine-grained analysis of affect dynamics during student's interaction with a human teacher, the use of multimodal signals in truly dynamic settings, and demonstration of the utility of the proposed approach to automatically detect behaviors and predict emotions.

We consider multiple thrusts of future work. With the current data set, we envision the following tasks worth consideration: (1) Learn latent dynamic model for problem solving state recognition so that it can be used to improve predictive model of ITC; (2) Explore the possibility of automatic transcription with Automatic Voice Recognition system, and explore sentiment analysis of mom's response; (3) Explore the utility of prosody features of speech signals to complement the current visual-cues based affect detection. Another research direction involves extending this study to more subjects so that inter-subject variation can be observed and modeled. In addition, we would also like to explore the possibility of transferring models learned from one child to another. It is also of interest to explore the correlation between metrics gathered in this study with psychological instruments such as grit scales [6]. Last but not least, we envision our current work to be a foundation for a future tool to guide teachers towards optimal interactions with their students in real time.

8. ACKNOWLEDGMENTS

We would like to thank Liangke Gui for help extracting features from FACET and COVARAP and Tadas Baltrusaitis for helping with Openface features. This work has been partially supported by NSF (1320347).

9. REFERENCES

[1] T. Baltrušaitis, P. Robinson, and L. P. Morency. OpenFace: an open source facial behavior analysis toolkit. *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2016.

[2] J. Beck. Engagement tracing: using response times to model student disengagement. *Proceeding of the 2005 conference on Artificial ...*, pages 88–95, 2005.

[3] H. Brugman and A. Russel. Annotating Multimedia/ Multi-modal resources with ELAN. *Proceedings of LREC 2004, Fourth International Conference on Language Resources and Evaluation*, 2004.

[4] G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer. COVAREP âĂŞ A COLLABORATIVE VOICE ANALYSIS REPOSITORY FOR SPEECH TECHNOLOGIES Computer Science Department , University of Crete , Heraklion , Greece Phonetics and Speech Laboratory , Trinity College Dublin , Ireland TCTS Lab - University of Mons , Belgium A. pages 960–964, 2014.

[5] S. K. D'Mello and A. Graesser. Multimodal semi-automated affect detection from conversational cues, gross body language, and facial features. *User Modeling and User-Adapted Interaction*, 20(2):147–187, 2010.

[6] A. L. Duckworth, C. Peterson, M. D. Matthews, and D. R. Kelly. Grit: perseverance and passion for long-term goals. *Journal of personality and social psychology*, 92(6):1087–1101, 2007.

[7] E. S. Edgington. *Randomization Tests*. Marcel Dekker, Inc., 1986.

[8] K. Forbes-riley. Predicting emotion in spoken dialogue from multiple knowledge sources. *Proceedings of the 4th Meeting of the North American Chapter of the Association for Computational Linguistics: : Human Language Technologies,*, pages 201–208, 2004.

[9] P. Gorge. *How to Solve It*. Princeton University Press, 1945.

[10] A. Graesser, Y. Ozuru, and J. Sullins. What is a good question? In M. G. McKeown & L. Kucan, editor, *Threads of coherence in research on the development of reading ability*, pages 112–141. Guilford, New York, New York, USA, 2009.

[11] I. Jraidi, M. Chaouachi, and C. Frasson. A hierarchical probabilistic framework for recognizing learners' interaction experience trends and emotions. *Advances in Human-Computer Interaction*, 2014, 2014.

[12] A. Kapoor, S. Mota, and R. W. Picard. Towards a Learning Companion that Recognizes Affect. *AAAI Fall symposium*, (543):2–4, 2001.

[13] B. Mc, S. D'Mello, B. King, P. Chipman, K. Tapp, and A. Graesser. Facial Features for Affective State Detection in Learning Environments. *29th Annual meeting of the cognitive science society*, pages 467–472, 2007.

[14] H. J. M. Pennings, J. van Tartwijk, T. Wubbels, L. C. a. Claessens, A. C. van der Want, and M. Brekelmans. Real-time teacher-student interactions: A Dynamic Systems approach. *Teaching and Teacher Education*, 37:183–193, 2014.

[15] J. Whitehill, Z. Serpell, Yi-Ching Lin, A. Foster, and J. R. Movellan. The Faces of Engagement: Automatic Recognition of Student Engagementfrom Facial Expressions. *IEEE Transactions on Affective Computing*, 5(1):86–98, 2014.

Joint Discovery of Skill Prerequisite Graphs and Student Models

Yetian Chen[⋆†], José P. González-Brenes[†], Jin Tian[⋆]

[⋆]Computer Science Department
Iowa State University
Ames, IA, USA
{yetianc, jtian}@iastate.edu

[†]Advance Computing and Data Science Lab
Pearson
San Diego, CA, USA
jose.gonzalez-brenes@pearson.com

ABSTRACT

Skill prerequisite information is useful for tutoring systems that assess student knowledge or that provide remediation. These systems often encode prerequisites as graphs designed by subject matter experts in a costly and time-consuming process. In this paper, we introduce *Combined student Modeling and prerequisite Discovery* (COMMAND), a novel algorithm for jointly inferring a prerequisite graph and a student model from data. Learning a COMMAND model requires student performance data and a mapping of items to skills (Q-matrix). COMMAND learns the skill prerequisite relations as a Bayesian network (an encoding of the probabilistic dependence among the skills) via a two-stage learning process. In the first stage, it uses an algorithm called Structural Expectation Maximization to select a class of equivalent Bayesian networks; in the second stage, it uses curriculum information to select a single Bayesian network. Our experiments on simulations and real student data suggest that COMMAND is better than prior methods in the literature.

Keywords

Prerequisite discovery, Bayesian network, student modeling

1. INTRODUCTION

Course *curricula* are usually organized in a meaningful sequence that evolves from relatively simple lessons to more complex ones. Among these lessons, some are required to be mastered by the student before the subsequent ones can be learned. For instance, students have to know how to do addition before they learn to do multiplication. We refer to *prerequisite structure* as the relationships among skills that place strict constraints on the order in which skills can be acquired.

Prerequisite structures are crucial for designing intelligent tutoring systems that assess student knowledge or that offer remediation interventions to students. Building such systems require prerequisite information that is often hand-engineered by subject matter experts in a costly and time-consuming process. Moreover, the prerequisite structures specified by the experts are seldom tested and might be unreliable in the sense that experts may have "blind spots".

Recent interest in computer assisted education promises large amounts of data from students solving *items*— questions, problems, parts of questions. Performance data –what items a learner answers correctly– can be used to create *student models*. These models represent an estimate of skill proficiency at a given point in time [17]. For example, a student model can represent that Alice has already mastered integer addition, but Bob has not. Student models are often used to personalize instruction in tutoring systems or to predict future student performance. In this paper, we introduce *Combined student Modeling and prerequisite Discovery* (COMMAND), a novel algorithm for simultaneously discovering prerequisite structure of skills and a student model from student performance data.

2. RELATION TO PRIOR WORK

Prior work has investigated how to discover prerequisites among items without considering their mapping into skills [6, 19]. Item-to-skill mappings (also called Q-matrices) are desirable because they allow more interpretable diagnostic information. Because of this, follow-up work [2, 4] has studied whether a pair of skills have a prerequisite relationship or not. For this, we can measure if a model that assumes a dependency between the two skills explains the data better than a model that assumes independence. This comparison can be done with data likelihood [2] or association rule mining [4]. Although promising, prior methods have limitations that we address:

1. We estimate the global prerequisite structure, not just the pairwise relationships. For example, suppose we want to discover the prerequisites of three skills for English learning (S_1:syntax, S_2:cohesion and S_3:lexical rules). If we use prior methods, we discover that the three skills are related among each other. However, pairwise methods are unable to tell if the relationships are due to indirect (e.g, $S_3 \to S_2 \to S_1$), or direct (e.g, $S_3 \to S_2 \to S_1$) effects.
2. It is unclear how to use the output of these prerequisite structures for student modeling. For example, it is not obvious how to best use them to make predictions of future student performance.
3. Prior work does not provide quantitative evaluation using real student data. Overall, learner data has been used to provide examples, but without any methodology that can help compare what algorithm works better.

A statistical formalism called Bayesian network has been useful to model prerequisite structures [12]. Bayesian networks allows modeling the full structure of skills (beyond pairwise relationships)

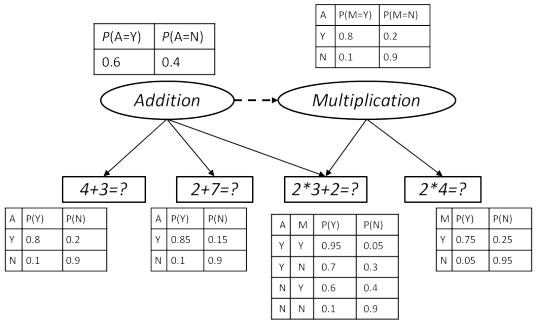


Figure 1: A hypothetical Bayesian network. Solid edges are given by item to skill mapping, dashed edges between skill variables are to be discovered from data. The conditional probability tables are to be learned.

and can encode conditional independence between the skills. Unfortunately, prior work with Bayesian networks requires a domain expert to design the prerequisite structures [10], and automatic techniques have not been demonstrated with real student data [14]. We now describe the COMMAND algorithm that discovers a Bayesian network that encodes the prerequisite structure of skills.

3. THE COMMAND ALGORITHM

COMMAND learns the prerequisite structure of the skills from data with a statistical model called Bayesian network [13, 15]. Bayesian networks are one type of probabilistic graphical models because they can be represented visually and algebraically as a collection of nodes and edges. A tutorial description of Bayesian networks in education can be found elsewhere [12], but for now we say that they are often described with two components: the nodes represent the random variables, which we describe using *conditional probability tables* (CPTs), and the set of edges that form a *directed acyclic graph* (DAG) represent the conditional dependencies between the variables. Bayesian networks are a flexible tool that can be used to model an entire curriculum.

Figure 1 illustrates an example of a prerequisite structure modeled with a Bayesian network. Here, we relate four test items with the skills of addition and multiplication. Addition is a prerequisite of multiplication thus there is an arrow from addition to multiplication. Modeling prerequisites as edges in a Bayesian network allows us to frame the discovery of the prerequisite relationships as the well-studied machine learning problem of learning a Bayesian network from data with the presence of unobserved latent variables. We represent the prerequisite structure using Bayesian networks that use latent binary variables to represent the student knowledge of a skill (i.e., mastery or not mastery), and observed binary variables that represent the student performance answering items (i.e., correct or incorrect).

Algorithm 1 describes the COMMAND pipeline. The input to COMMAND is a matrix $\mathbf{D}$ with $n \times p$ dimensions, representing n students, answering p items. Each entry in $\mathbf{D}$ encodes the performance of a student (see Table 1 for an example). Additionally, we require a Q-matrix to represent the item-to-skill mapping. Q-matrices are often designed by subject matter experts but automatic methods to discover them exist [8].

Table 1: Example student performance matrix to use with COMMAND. The performance of a student is encoded with 1 if the student answered correctly the item, and 0 otherwise.

User	Item 1	Item 2	Item 3	Item p
Alice	0	1		0
Bob	1	1	...	1
Carol	0	0		1
		...		

Algorithm 1 The COMMAND algorithm

Require: A matrix $\mathbf{D}$ of student performance on a set of test items, skill-to-item mapping Q (containing a set of skills $\mathbf{S}$).

1: $G_0 \leftarrow \text{Initialize}(\mathbf{S}, Q)$
2: $i \leftarrow 0$ } Initialization
3: **do**
4: E-step:
5: $\Theta_i^* \leftarrow \text{ParametricEM}(G_i, \mathbf{D})$
6: $\mathbf{D}_i^* \leftarrow \text{Inference}(G_i, \Theta_i^*, \mathbf{D})$
7: M-step:
8: $\langle G_{i+1}, \Theta_{i+1} \rangle \leftarrow \text{BNLearning}(G_i, \mathbf{D}_i^*)$
9: $i \leftarrow i+1$ } Structural EM
10: **while** stop criterion is not met
11: $RE \leftarrow FindReversibleEdges(G_i)$
12: $EC \leftarrow EnumEquivalentDAGs(G_i)$
13: $DE \leftarrow \{\}$
14: **for** every reversible edge $S_i - S_j$ in RE **do**
15: $ratio \leftarrow \frac{P(S_j=0|S_i=0)}{P(S_i=0|S_j=0)}$[1]
16: **if** $ratio \geq 1$ **then**
17: $ratio^* = ratio$
18: $DE \leftarrow DE \cup S_i \rightarrow S_j$
19: **else**
20: $ratio^* = \frac{1}{ratio}$
21: $DE \leftarrow DE \cup S_i \leftarrow S_j$ } Discriminate between equivalent BNs
22: **end if**
23: **end for**
24: $sort(DE)$ by $ratio^*$ in descending order
25: **while** DE is not empty **do**
26: $e \leftarrow dequeue(DE)$
27: **if** $\exists G \in EC\ e \in G$ **then**
28: $\forall G \in EC$, remove G from EC if $e \notin G$
29: **end if**
30: **end while**
31: return EC

COMMAND relies on a popular machine learning algorithm called *Structural Expectation Maximization* (Structural EM), which to the extent of our knowledge has not been used in educational applications before. Structural EM extends the Expectation Maximization (EM) algorithm to allow efficient structure learning of Bayesian networks when there are latent variables or missing values in the data. A secondary contribution of our work is introducing Structural EM for learning Bayesian network structures from educational data. We now describe the steps of COMMAND in detail.

3.1 Initial Bayesian Network

COMMAND first creates an initial Bayesian network using the Q-matrix by creating an arc to each item from each of its required

[1] $P(S_i = a|S_j = b)$ can be computed using any Bayesian network inference algorithm such as Junction tree algorithm [11].

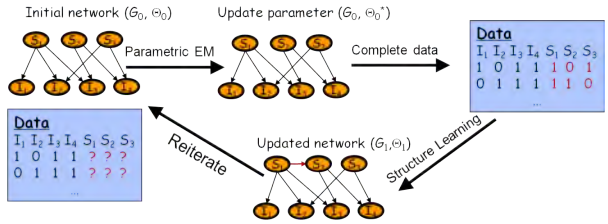


Figure 2: An illustration of the Structure EM algorithm to discover the structure of the latent variables. G represents the DAG structure. Θ is the set of conditional probability tables (CPTs).

skills. Because there are no edges between the skills, this initial network does not encode any prerequisite information. COMMAND uses Structural EM to learn arcs (prerequisites) between the skill variables.

3.2 Structural EM

A common solution to learning a Bayesian network from data is the score-and-search approach [5, 9]. This approach uses a scoring function (like the Bayesian Information Criterion (BIC)) to measure the fitness of a Bayesian network structure to the observed data, and it attempts to find the optimal model in the space of all possible Bayesian network structures. However, the conventional score-and-search approaches rely on efficient computation of the scoring function, which is only feasible for problems where data contain observations for all variables in the Bayesian network. Unfortunately, our domain has skill variables that are not directly observed. An intuitive work-around is to use EM to estimate the scoring function. However, in this case EM takes a large number (hundreds) of iterations that require Bayesian network inference, which is computationally prohibitive. Further, we need run EM for each candidate structure, and the number of possible Bayesian network structures is super-exponential with respect to the number of nodes. The Structural EM algorithm [7] is an efficient alternative.

Structural EM is an iterative algorithm that inputs a matrix **D** of student performance (see example Table 1). Figure 2 illustrates one iteration of the Structural EM algorithm. The relevant steps are also sketched in Algorithm 1. Each iteration consists of an Expectation step (*E-step*) and a Maximization step (*M-step*). In the *E-step*, it first finds the maximum likelihood estimate Θ^* of the CPTs for the current structure G calculated from previous iteration using parametric EM. It then does Bayesian inference to compute the expected values for the latent variables using the current model (G, Θ^*), and uses the values to complete the data. In the *M-step*, it uses the conventional score-and-search approach to optimize the structure according to the completed data (as if the latent variables were observed). Since the space of possible Bayesian network structures is super-exponential, exhaustive search is intractable and local search algorithms, such as greedy hill-climbing search, are often used. The *E-step* and *M-step* interleave and iterate until some stop criterion is met, e.g., the scoring function does not change significantly. Contrast to the conventional score-and-search algorithm, Structural EM runs EM only on one structure in each iteration, thus is computationally more efficient.

We use an efficient implementation of Structural EM available online called LibB[2]. Because COMMAND's initialization step fixes the arcs from skills to items according to the Q-matrix, the *M-step* only needs to consider the candidate structures that comply with the Q-matrix. An advantage of using Structural EM to discover the prerequisite relationship of skills is that it can be easily extended to incorporate domain knowledge. For example, we can place constraints on the output structure to force or to disallow a skill to be a prerequisite of another skill. Another advantage of Structural EM is that it can be applied when there are missing data in the student performance matrix **D** [7]. That is, some students do not answer all the items. The general idea is, in the *E-step*, the algorithm also computes the expected values for missing data points, in addition for latent variables.

[2] `http://compbio.cs.huji.ac.il/LibB/programs.html`

3.3 Discriminate Between Equivalent BNs

Structural EM selects a Bayesian network model based on how well it explains the distribution of the data. Bayesian network theory states that some Bayesian networks are statistically equivalent in representing the data. Thus, the output from Structural EM is actually an equivalence class (EC) that may contain many Bayesian network structures[3]. These equivalent Bayesian networks have the same skeleton and the same v-structures[4]. For instance, Figure 3 gives an example of a simple equivalence class containing three Bayesian networks that are not distinguishable by Structural EM algorithm and the method in [14]. They share the skeleton but differ in the orientation of at least one of the edges (we will call such an edge a reversible edge). They apparently represent three different prerequisite structures.

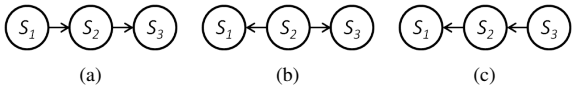


Figure 3: Three equivalent Bayesian networks representing different prerequisite structures.

3.3.1 Domain Knowledge

To determine a unique structure, we use a heuristic based in domain knowledge to determine the orientation of each reversible edge. For convenience in notation, let's assume that the random variables that represent skill proficiency can take two values: 0 if the skills is not mastered, and 1 if the skill is mastered. Our assumption is that if a skill S_1 is the prerequisite of a skill S_2, a student can not master skill S_2 before she masters S_1. More formally:

Assumption. If S_1 is a prerequisite of S_2 (i.e., $S_1 \rightarrow S_2$), then $S_1 = 0 \Rightarrow S_2 = 0$. In other words, $P(S_2 = 0|S_1 = 0) = 1$.

Our assumption implies that S_1 cannot be a prerequisite of S_2 if $P(S_2 = 0|S_1 = 0) = 1$ does not hold. This puts a constraint on the joint distribution encoded by the Bayesian network to be learned.

For example, consider the case of choosing the orientation of a reversible edge $S_1 - S_2$ from $S_1 \leftarrow S_2$ or $S_1 \rightarrow S_2$. We can check whether $P(S_2 = 0|S_1 = 0) = 1$ or $P(S_1 = 0|S_2 = 0) = 1$. However, it is possible that our assumption does not hold, and a student got to master a skill even if he does not know the prerequisite. Moreover, because of statistical noise, the conditional probability $P(S_2 = 0|S_1 = 0)$ may not be exactly 1. Thus, we use the following empirical rule:

[3] Structural EM outputs a DAG. However, the scoring function does not discriminate between the many DAGs of the equivalence class.

[4] A v-structure with nodes u, v, w in a DAG are the directed edges $u \rightarrow v$ and $w \rightarrow v$ and u and w are not adjacent in the DAG [18].

Rule 1. if $P(S_2 = 0|S_1 = 0) \geq P(S_1 = 0|S_2 = 0)$, we determine $S_1 \rightarrow S_2$; otherwise, we determine $S_1 \leftarrow S_2$.

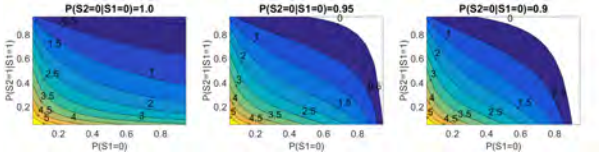
Note that these two conditional probabilities can be computed easily from the Bayesian network model output from Structural EM. The intuition behind this rule is that the conditional probability $P(S_2 = 0|S_1 = 0)$ can be interpreted as the strength of the prerequisite relationship $S_1 \rightarrow S_2$. The larger of this probability, the more likely the relationship $S_1 \rightarrow S_2$ holds. Since here we are concerned with which direction the edge goes, we simply compare the two probabilities and select the direction that is more probable. Note that $P(S_2 = 0|S_1 = 0) = 1$ and $P(S_1 = 0|S_2 = 0) = 1$ may hold simultaneously. If $S_1 \rightarrow S_2$ is true, $P(S_1 = 0|S_2 = 0) = 1$ only if $P(S_1 = 1) = 0$ or if $P(S_2 = 0|S_1 = 1) = 0$.[5] If $P(S_1 = 1) = 0$, this implies that no student knows S_1. If $P(S_2 = 0|S_1 = 1) = 0$, it means that learning S_2 becomes trivial once students know S_1. For simplicity, we ignore this extreme case.

3.3.2 Theoretical Justification of Heuristic

We now provide theoretical justification for the rule we propose. Consider a simple equivalence class, which contains two equivalent DAGs $S_1 \rightarrow S_2$ and $S_1 \leftarrow S_2$, where the true model is $S_1 \rightarrow S_2$. We have three free conditional probability parameters: $P(S_1 = 0) = p$, $P(S_2 = 0|S_1 = 0) = q$, $P(S_2 = 1|S_1 = 1) = r$. Let's define a *ratio* that quantifies choosing the true model:

$$ratio = \frac{P(S_2 = 0|S_1 = 0)}{P(S_1 = 0|S_2 = 0)}. \quad (1)$$

Using Bayes rule and rules of probability, the rule $ratio \geq 1$ becomes $(1-p)(1-r) - p(1-q) \geq 0$. Since *ratio* depends on p, q and r, we study how *ratio* changes with these parameters. Figure 4 shows the contour plots of $log(ratio)$ against $P(S_1 = 0)$ and $P(S_2 = 1|S_1 = 1)$ for three different values of $P(S_2 = 0|S_1 = 0)$. The white region in each contour plot is the region where our heuristic fails because $ratio < 1$. Figure 4(a) shows that when $P(S_2 = 0|S_1 = 0) = q = 1$, our heuristic rule is always correct, no matter what, because there is no white space. With $P(S_2 = 0|S_1 = 0)$ decreasing, the white region becomes larger and the rule becomes less accurate. As mentioned, $P(S_2 = 0|S_1 = 0)$ can be interpreted as the strength of the prerequisite relationship. If we fix the value of $P(S_2 = 0|S_1 = 0)$ and assume that the two free parameters p and r are independent and uniformly distributed, then the area of the white region represents the probability that the rule makes a wrong decision. As the strength of the prerequisite relationship gets weaker, our rule to determine the prerequisite relationship becomes less accurate.



(a) (b) (c)

Figure 4: Contour plots of $log(ratio)$ against $P(S_1 = 0)$ and $P(S_2 = 1|S_1 = 1)$ for various values of $P(S_2 = 0|S_1 = 0)$.

3.3.3 Orient All Reversible Edges

Using our proposed rule, we can orient every reversible edge in the network structure. However, orienting each reversible edge is not independent and may conflict with each other. Having oriented one edge would constrain the orientation of other reversible edges because we have to ensure the graph is a DAG and the equivalence property is not violated. For example, in Figure 5a, if we have determined $S_1 \rightarrow S_2$, the edge $S_2 \rightarrow S_3$ is enforced. In this paper, we take an ad-hoc strategy to determine the orientation for all reversible edges. For each reversible edge $S_i - S_j$, we let $ratio^* = ratio$ if $ratio \geq 1$ and $ratio^* = \frac{1}{ratio}$ otherwise. The larger the $ratio^*$ is, the more confidently when we decide the orientation. We sort the list of reversible edges by $ratio^*$ in descending order. We then orient the edges by this ordering. In our implementation, we use the following strategy: we first enumerate all equivalent Bayesian networks and make them a list of candidates; when an edge is oriented to $S_i \rightarrow S_j$, we remove all contradicting Bayesian networks from the list. Eventually only one Bayesian network structure stands. This procedure is detailed in the *Discriminate between equivalent BNs* section of Algorithm 1. The $EnumEquivalentDAGs(G_i)$ implements the algorithm of enumerating equivalent DAGs in [3].

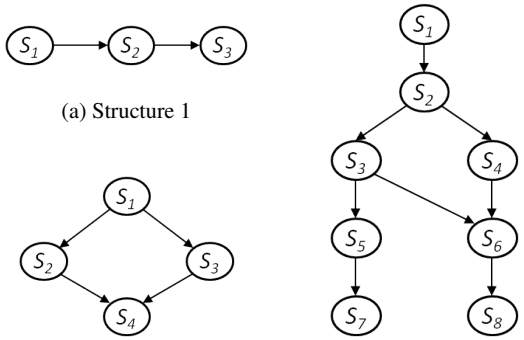
[5] Since $P(S_1 = 0|S_2 = 0) = \frac{P(S_2=0|S_1=0)P(S_1=0)}{P(S_2=0|S_1=0)P(S_1=0)+P(S_2=0|S_1=1)P(S_1=1)}$, $P(S_1 = 0|S_2 = 0) = 1$ only if $P(S_2 = 0|S_1 = 1)P(S_1 = 1) = 0$.

4. EVALUATION

In § 4.1, we evaluate COMMAND with simulated data to assess the quality of the discovered prerequisite structures. Then, in § 4.2 we use data collected from real students. In all our experiments, we use BIC as the scoring function in Structural EM .

4.1 Simulated Data

Synthetic data allow us to study how COMMAND compares to the ground truth. For this, we engineered three prerequisite structures (DAGs), shown in Figure 5. Here, each figure represents different causal relations between the simulated latent skill variables.



(a) Structure 1

(b) Structure 2 (c) Structure 3

Figure 5: Three different DAGs between latent skill variables. Item nodes are omitted.

For clarity, Figure 5 omits the item nodes; but each skill node is parent of six item variables and each item variable has 1-3 skill nodes as parents. All of these nodes are modeled using binary random variables. More precisely, the latent nodes represent whether the student achieves mastery of the skill, and the observed nodes indicate if the student answers the item correctly. Notice that these networks include the prerequisite structures as well as the skill-item mapping.

We consider simulated data with different number of observations ($n = 150, 500, 1000, 2000$). For each sample size and each DAG, we generate ten different sets of conditional probability tables randomly with three constraints. First, we enforce that achieving mastery of the prerequisites of a skill will increase the likelihood of mastering the

skill. Second, for each prerequisite pair $S_i \rightarrow S_j$, $P(S_j = 0|S_i = 0)$ is randomly selected to be in $[0.9, 1.0]$. Finally, mastery of a skill increases the probability of student correctly answering the test item. In total we generated 120 synthetic datasets (3 DAGs x 4 sample sizes x 10 CPTs), and report the average results.

We evaluate how well COMMAND can discover the true prerequisite structure using metrics designed to evaluate Bayesian networks structure discovery. In particular, we use the F_1 *adjacency score* and the F_1 *orientation score*. The adjacency score measures how well we can recover connections between nodes. It is a weighted average of the true positive adjacency rate and the true discovery adjacency rate. On the other hand, the orientation score measures how well we can recover the direction of the edges. It is calculated as a weighted average of the true positive orientation rate and true discovery orientation rate. In both cases, the F_1 score reaches its best value at 1 and worst at 0. Moreover, for comparison, we compute the F_1 *adjacency score* for Bayesian network structures whose skill nodes are fully connected with each other. These fully connected DAGs will serve as baselines for evaluating the adjacency discovery[6]. For completeness, we list these formulas in tables 2 and 3, respectively.

Table 2: Formulas for measuring adjacency rate (AR)

Metric	Formula
True positive (*TPAR*)	$\frac{\text{\# of correct adjacencies in learned model}}{\text{\# of adjacencies in true model}}$
True discovery (*TDAR*)	$\frac{\text{\# of correct adjacencies in learned model}}{\text{\# of adjacencies in learned model}}$
F_1-*AR*	$\frac{2 \cdot TPAR \cdot TDAR}{TPAR+TDAR}$

Table 3: Formulas for measuring orientation rate (OR)

Metric	Formula
True positive (*TPOR*)	$\frac{\text{\# of correctly directed edges in learned model}}{\text{\# of directed edges in true model}}$
True discovery (*TDOR*)	$\frac{\text{\# of correctly directed edges in learned model}}{\text{\# of directed edges in learned model}}$
F_1-*OR*	$\frac{2 \cdot TPOR \cdot TDOR}{TPOR+TDOR}$

We use these metrics to evaluate the effect of varying the number of observations of the training set (sample size) on the quality of learning the prerequisite structure. We designed experiments to specifically answer the following four questions:

1. How does the type of items affect COMMAND's ability to recover the prerequisite structure? We consider the situation where in the model each item requires only one skill and the situation where each item requires multiple skills.
2. How well does COMMAND perform when there is noise in the data? We focus on studying noise due to the presence of unaccounted latent variables.
3. How well does COMMAND perform when the student performance data have missing values?
4. How is COMMAND compared with other prerequisite discovery methods? In particular, we compare COMMAND to the Probabilistic Association Rules Mining (PARM) method [4].

We now investigate these questions.

[6]We do not compute F_1 orientation score for fully connected DAGs because all edges in a fully connected DAG are reversible.

4.1.1 Single-skill vs Multi-skill Items

We consider two situations where different types of Q-matrix are used. In the first situation, each item node maps to exactly one skill node. In the second one, each item maps to 1-3 skills. Figure 6 compares the F_1 of adjacency discovery and edge orientation results under the two types of Q-matrices. With only 500 observations, COMMAND improves on a fully connected Bayesian network baseline. COMMAND's accuracy improves with the amount of data, but its accuracy is slightly lower when the Q-matrix contains items that require more than one skill. A possible explanation for this is that multi-skill items may introduce more spurious correlations in the data. With just 2000 observations, COMMAND recovers the true structures almost perfectly.

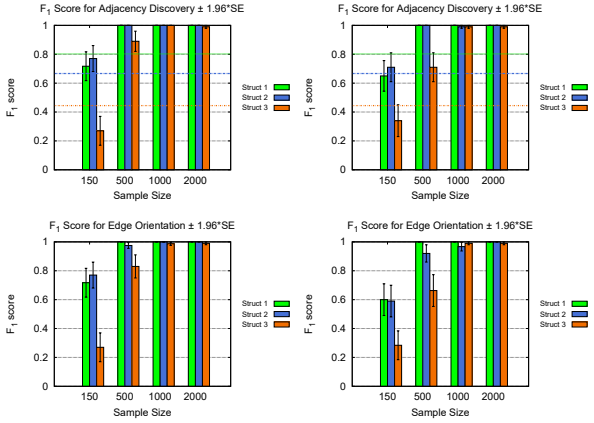


Figure 6: Comparison of F_1 scores for adjacency discovery (top row) and for edge orientation (bottom row). Horizontal lines are baseline scores for fully-connected (complete) networks. The error bars show the 95% confidence intervals, i.e., $\pm 1.96 * \text{SE}$.

4.1.2 Sensitivity to Noise

Real-world data sets often contain various types of noise. For example, noise may occur due to latent variables that are not explicitly modeled. To evaluate the sensitivity of COMMAND to noise, we synthesize the three Bayesian networks in Figure 5 to include a *StudentAbility* node that takes three possible states (low/med/high). In these Bayesian networks, students' performance depends not only on whether they have mastered the skills, but also on their individual ability. For simplicity, all items in the setting are single-skilled items. We first simulated data from Bayesian networks that have a *StudentAbility* variable to generate "noisy" data samples, and then use this data to recover the prerequisite structure. Figure 7 illustrates the procedure of this sensitivity analysis experiment for Structure 1.

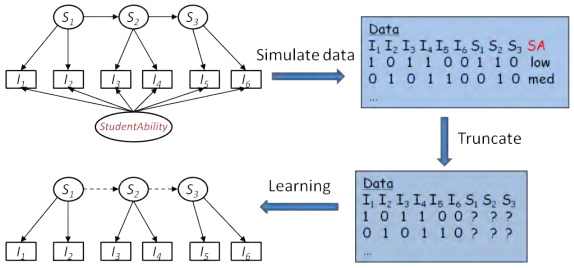


Figure 7: Evaluation of COMMAND with noisy data

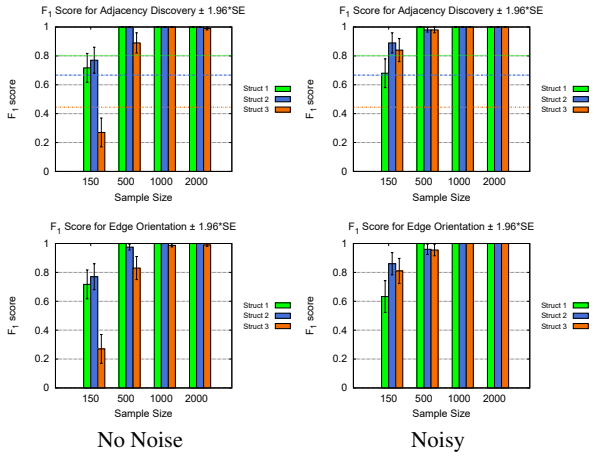


Figure 8: Results of adding systematic noise. Top: Comparison of F_1 scores for adjacency discovery. Horizontal lines are baseline F_1 scores computed for fully connected Bayesian networks. Bottom: Comparison of F_1 scores for edge orientation.

Figure 8 compares the results where noise was introduced or not. Interestingly, the noise actually improves COMMAND's accuracy. This improvement is more evident when the sample size is small (see $n = 150$). For smaller sample sizes, Structural EM usually discovers less relationships than actually exist, because BIC prefers sparse structures. We hypothesize that the correlations caused by *StudentAbility* node would cause Structural EM to add "stronger" edges between skill nodes, resulting in higher F1.

4.1.3 Sensitivity to Missing Values

Real-world datasets collected from students often have missing values, for example, when learners do not answer all items. To evaluate how COMMAND performs on data with missing values, we generated data sets of with 1000 observations with varying fraction of randomly missing values (10%, 20%, 30%, 40%, 50%). We used COMMAND to recover the structures from these data sets. Again, the models only contain single-skilled items. Figure 9 shows the results of this experiment. Although accuracy decreases when the fraction of missing values increases, COMMAND is able to recover the true structures for Structure 1 and 2 even when the data contain up to 30% missing values.

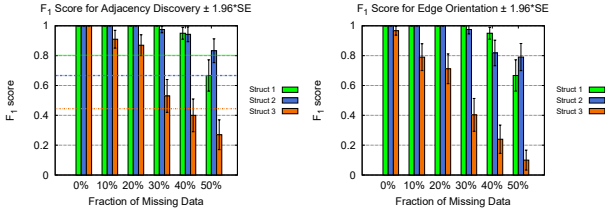


Figure 9: Results of learning with missing data. Left: Comparison of F_1 scores for adjacency discovery. Horizontal lines are baseline F_1 scores computed for fully connected Bayesian networks. Right: Comparison of F_1 scores for edge orientation.

4.1.4 Comparison With Prior Work

The Probabilistic Association Rules Mining (PARM) is a recent algorithm for discovering the prerequisite relationships between skills [4]. In this approach, a prerequisite relationship $S_1 \rightarrow S_2$ is considered to exist if $P(S_1 = 1, S_2 = 1) \geq minsup \wedge P(S_1 = 1 | S_2 = 1) \geq minconf) \geq minprob$ and $P(P(S_1 = 0, S_2 = 0) \geq minsup \wedge P(S_2 = 0 | S_1 = 0) \geq minconf) \geq minprob$, where $minsup$, $minconf$ and $minprob$ are pre-specified constants between 0 and 1.

We simulate data from Structure 3 from Figure 5(c) (with single-skilled items), which has 21 pair-wise prerequisite relationships. We derive pair-wise prerequisite relationships from this network and see how the two approaches discover these relationships. When experimenting with PARM, we use $minsup = 0.125$, $minconf = 0.76$, $minprob = 0.9$, because they were suggested by the authors [4].

PARM is limited to discovering pair-wise prerequisite relationships (instead of constructing the full structure). To make a fair comparison, we evaluate how accurately COMMAND and PARM discover relationship pairs. For this, we use the F1 metric in Table 2, but we count pairs of *related skills* instead of adjacencies. Two skills are related if one is a descendant of the other one. Figure 10 shows that COMMAND outperforms PARM, and the difference becomes significant for sample size $n \geq 500$. The low F_1 score of by PARM is because it fails to discover many prerequisite relationships (data not shown), and because PARM does not respect transitivity. For example, PARM may reject $S_1 \rightarrow S_3$ even it has discovered $S_1 \rightarrow S_2$ and $S_2 \rightarrow S_3$. We speculate that selecting a different set of cutoff values for PARM may improve the results. However, determining these thresholds is not trivial and may require experts' intervention. By contrast, COMMAND does not require tuning.

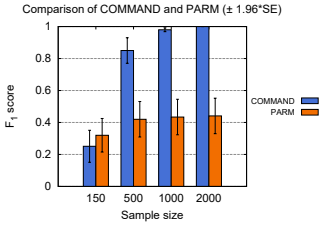


Figure 10: Comparison of COMMAND and PARM for discovering prerequisite relationships in Structure 3.

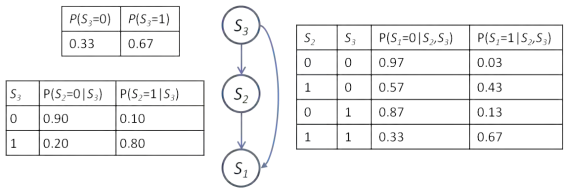
4.2 Real Student Performance Data

We now evaluate COMMAND using two real-world data sets.

4.2.1 English Data Set

The Examination for the Certification of Proficiency in English (ECPE) dataset describes 2922 examines in their understanding of English language grammar [16]. The dataset includes student performance in 28 items on 3 skills (S_1: morphosyntactic rules, S_2: cohesive rules, and S_3:lexical rules). Each item requires either one or two of the three skills.

Figure 11 shows the prerequisite structure discovered with COMMAND. It hypothesizes that lexical rules is a prerequisite of cohesive rules and morphosyntactic rules; cohesive rules is a necessary skill for learning morphosyntactic rules. The pair-wise prerequisite relationships totally agrees with the findings in [16] and that by the PARM method in [4]. Our model infers a complete DAG, suggesting that there are no conditional independencies among the three



$P(S_3=0)$	$P(S_3=1)$
0.33	0.67

S_3	$P(S_2=0 \mid S_3)$	$P(S_2=1 \mid S_3)$
0	0.90	0.10
1	0.20	0.80

S_2	S_3	$P(S_1=0 \mid S_2,S_3)$	$P(S_1=1 \mid S_2,S_3)$
0	0	0.97	0.03
1	0	0.57	0.43
0	1	0.87	0.13
1	1	0.33	0.67

Figure 11: The estimated DAG and CPTs of the ECPE data set.

skills. This is an interesting insight that previous approaches cannot provide. Further, COMMAND also outputs the conditional probabilities associated with each skill and its direct prerequisite. We clearly see that the probability of student mastering a skill increases when the student has acquired more prerequisites of the skill.

4.2.2 Math Data Set

We now evaluate COMMAND using data collected from a commercial non-adaptive tutoring system. The textbook items are classified in chapters, sections, and objectives. We only use student performance data from tests in Chapter 2 and 3. That is, students are tested on the items after they have been taught all relevant skills.

Q-matrix and preprocessing. We define skills as book sections. We use a Q-matrix that assigns each exercise to a skill solely as the book section in which the item appears.[7] For each chapter, we process the data to find a subset of items and students that do not have missing values. That is, the datasets we use in COMMAND have students responding to *all* of the items.

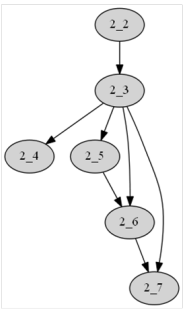
After filtering, two data sets, `Math-chap2` and `Math-chap3`, were obtained for Chapter 2 and 3 respectively. In `Math-chap2`, six skills are included and each skill is tested on three to eight items, for a total of 30 items. In `Math-chap3`, seven skills are included and each skill has three to seven items, for a total of 33 items. `Math-chap2` includes student test results for 1720 students, while the `Math-chap3` has test results for 1245 students. For simplicity we use binary variables to encode performance data and skill variables.

Prerequisite Structure Discovery. The Bayesian networks generated with the COMMAND algorithm are illustrated in Figure 12. Our observation is that the topological order of the sections in both structures are fully consistent with the book ordering heuristic. This shows an agreement between our fully data-driven method and human experts. We also ran PARM approach to learn pair-wise prerequisite relationships from these data sets. Given $minsup = 0.125$, $minconf = 0.76$ and $minprob = 0.9$, 2_5 $\rightarrow$ 2_6, 2_5 $\rightarrow$ 2_7 and 2_6 $\rightarrow$ 2_7 are discovered for `Math-chap2`, 3_1 $\rightarrow$ 3_3 and 3_2 $\rightarrow$ 3_3 are discovered for `Math-chap3`. These relationships are small subset of the set of relationships discovered by COMMAND.

Predictive Performance. COMMAND outputs a Bayesian network model that can be used for inference and predictive modeling. For example, given a student's response to a set of items, we can infer the student's knowledge status of a skill. We could use COMMAND to identify students that may need remediation because they

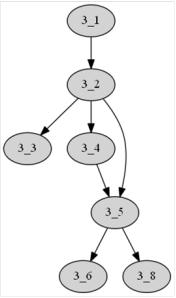
[7]Here we assume the items are single-skilled despite that they might be multi-skilled.

Skill ID	Skill Name
2_2	Symbols and Sets of Numbers
2_3	Fractions and Mixed Numbers
2_4	Exponents, Order of Operations, Variable Expressions, and Equations
2_5	Adding Real Numbers
2_6	Subtracting Real Numbers
2_7	Multiplying and Dividing Real Numbers



(a) Prerequisite structure learned for `Math-chap2`.

Skill ID	Skill Name
3_1	Simplifying Algebraic Expressions
3_2	The Addition and Multiplication Properties of Equality
3_3	Solving Linear Equations
3_4	An Introduction to Problem Solving
3_5	Formulas and Problem Solving
3_6	Percent and Mixture Problem Solving
3_8	Solving Linear Inequalities



(b) Prerequisite structure learned for `Math-chap3`.

Figure 12: Prerequisite structures constructed by COMMAND for `Math` data sets.

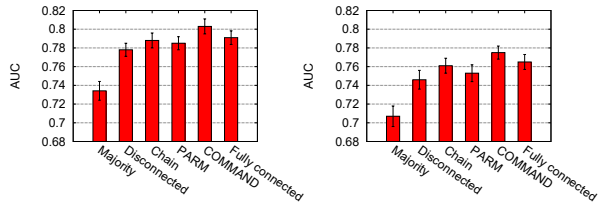
lack some background. We evaluate the accuracy of the predicted student performance on an item, when we observe the student response on the other items. More precisely, we compute the posterior probability of a student's response to an item I_i given his performance on all other items $\mathbf{I}_{-i} = \mathbf{I} \setminus \{I_i\}$, by marginalizing over the set of latent variables $\mathbf{S}$:

$$P(I_i|\mathbf{I}_{-i} = \mathbf{i}_{-i}) = \sum_{\mathbf{S}} P(I_i, \mathbf{S}|\mathbf{I}_{-i} = \mathbf{i}_{-i}).$$

This probability can be computed efficiently using the Junction tree algorithm [11]. We then do binary classification based on the posterior probability to determine if the student is likely to answer correct. We compare the Bayesian network models generated from COMMAND with five baseline predictors:

- A *majority* classifier which always classifies an instance to the majority class. For example, if majority of the students get an item wrong, other students would likely get it wrong.
- A Bayesian network model in which the skill variables are *disconnected*. This model assumes that the skill variables are marginally independent of each other. Most existing knowledge tracing approaches make this assumption.
- A Bayesian network model in which the skill variables are connected in a *chain* structure, i.e., 2-2$\rightarrow$2-3$\rightarrow$2-4$\rightarrow \ldots$ This assumes that a section (skill) only depends on the previous section. In other words, a first-order Markov chain dependency structure.
- A Bayesian network model constructed using the pairwise relationships output from *PARM*. That is, we create an edge $S_i \rightarrow S_j$ if PARM says S_i is the prerequisite of S_j.

- A *fully connected* Bayesian network where skill variables are fully connected with each other. This model assumes no conditional independence between skill variables and can encode any joint distribution over the skill variables. However, it has exponential number of free parameters and thus can easily overfit the data.



(a) `Math-chap2` AUC results. (b) `Math-chap3` AUC results.

Figure 13: Ten fold cross-validation results of evaluating the predictions of student performance.

The parameters of these baseline Bayesian network predictors are estimated from the data using parametric EM. The model predictions were evaluated using the *Area Under the Curve* (AUC) of the Receiver Operating Characteristic (ROC) curve metric calculated from 10-fold cross-validation. Results are presented in Figure 13. The error bars show the 95% confidence intervals calculated from the cross-validation. On both `Math-chap2` and `Math-chap3` data sets, the COMMAND models outperform the other five models. The *fully connected* models are the second best performing models. On `Math-chap2`, COMMAND model has an AUC of 0.803 ± 0.008 and the *fully-connected* model has an AUC of 0.791 ± 0.007 (Figure 13a). A paired t-test reveals that the AUC difference of two models are statistically significant with a p-value of 0.0022. On `Math-chap3`, COMMAND model has an AUC of 0.775 ± 0.007 and the *fully-connected* model has an AUC of 0.765 ± 0.008 (Figure 13b). The AUC difference of two models are also statistically significant with a p-value of 0.01. The *fully connected* models are outperformed by the much simpler prerequisite models, suggesting overfitting.

5. CONCLUSION AND DISCUSSION

Prerequisite graphs have been shown [1, 10] to improve student models. However, discovering the prerequisites between skills requires significant effort from subject matter experts. The main contribution of our work is a novel algorithm that simultaneously infers a prerequisite graph and a student model from data with less human intervention.

We extend on prior work in significant ways. We optimize the full structure of skills that captures the conditional independence between skills, instead of only estimating the pairwise relationships. Our experiments suggests that this results in better accuracy. Moreover, we argue that our strategy is easier to use because it does not require manual tuning of parameters. Other methods [2] require the *guess* and *slip* probabilities to be provided as input, or alternatively [4], thresholds to determine the existence of a prerequisite relationship. Determining these values requires experts' intervention. COMMAND does not require such tuning.

We analyze how missing values, noise and dataset size can affect the performance of COMMAND. Further research could explore additional datasets and baselines. A secondary contribution of our work is that we develop a methodology to evaluate prerequisite structures on real student data. We believe that we are the first to compare prerequisite discovery strategies by how well they can be used to predict student performance. Therefore, we validate COMMAND not only with synthetic data, but with two real-world datasets. Our results suggest that COMMAND improves on the state of the art because it significantly improves on a recently published technique.

Learning a prerequisite graph is not merely discovering a Bayesian network— equivalent Bayesian network structures in fact represent different prerequisite structures. We believe we are the first to address this problem. We use domain knowledge to refine the prerequisite models output using a theoretically motivated method.

6. REFERENCES

[1] Anthony Botelho, Hao Wan, and Neil Heffernan. 2015. The prediction of student first response using prerequisite skills. In *Learning At Scale*. ACM, 39–45.

[2] Emma Brunskill. 2010. Estimating prerequisite structure from noisy data. In *Educational Data Mining 2011*.

[3] Yetian Chen and Jin Tian. 2014. Finding the k-best Equivalence Classes of Bayesian Network Structures for Model Averaging. In *AAAI*. 2431–2438.

[4] Yang Chen, Pierre-Henri Wuillemin, and Jean-Marc Labat. 2015. Discovering Prerequisite Structure of Skills through Probabilistic Association Rules Mining. In *Educational Data Mining*. 117–124.

[5] Gregory F Cooper and Edward Herskovits. 1992. A Bayesian method for the induction of probabilistic networks from data. *Machine learning* 9, 4 (1992), 309–347.

[6] Michel C Desmarais, Peyman Meshkinfam, and Michel Gagnon. 2006. Learned student models with item to item knowledge structures. *User Modeling and User-Adapted Interaction* 16, 5 (2006), 403–434.

[7] Nir Friedman. 1997. Learning belief networks in the presence of missing values and hidden variables. In *ICML*, Vol. 97. 125–133.

[8] José P. González-Brenes. 2015. Modeling Skill Acquisition Over Time with Sequence and Topic Modeling. In *International Conference on Artificial Intelligence and Statistics*. 296–305.

[9] David Heckerman, Christopher Meek, and Gregory Cooper. 1997. *A Bayesian approach to causal discovery*. Technical Report. MSR-TR-97-05, Microsoft Research.

[10] Tanja Käser, Severin Klingler, Alexander Gerhard Schwing, and Markus Gross. 2014. Beyond knowledge tracing: Modeling skill topologies with bayesian networks. In *Intelligent Tutoring Systems*. Springer, 188–198.

[11] Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

[12] Robert J Mislevy, Russell G Almond, Duanli Yan, and Linda S Steinberg. 1999. Bayes nets in educational assessment: Where the numbers come from. In *Uncertainty in artificial intelligence*. 437–446.

[13] Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.

[14] Richard Scheines, Elizabeth Silver, and Ilya Goldin. 2014. Discovering prerequisite relationships among knowledge components. In *Educational Data Mining 2014*.

[15] Peter Spirtes, Clark Glymour, and Richard Scheines. 2001. *Causation, prediction, and search*. MIT Press.

[16] Jonathan Templin and Laine Bradshaw. 2014. Hierarchical diagnostic classification models: A family of models for estimating and testing attribute hierarchies. *Psychometrika* 79, 2 (2014), 317–339.

[17] Kurt VanLehn. 1988. Student modeling. *Foundations of intelligent tutoring systems* 55 (1988), 78.

[18] Thomas Verma and Judea Pearl. 1990. Equivalence and synthesis of causal models. In *Uncertainty in Artificial Intelligence*. 255–270.

[19] Annalies Vuong, Tristan Nixon, and Brendon Towle. 2010. A method for finding prerequisites within a curriculum. In *Educational Data Mining 2011*.

Gauging MOOC Learners' Adherence to the Designed Learning Path

Dan Davis*, Guanliang Chen†, Claudia Hauff and Geert-Jan Houben
Delft University of Technology
Delft, the Netherlands
{d.j.davis, guanliang.chen, c.hauff, g.j.p.m.houben}@tudelft.nl

ABSTRACT

Massive Open Online Course (MOOC) platform designs, such as those of edX and Coursera, afford linear learning sequences by building scaffolded knowledge from activity to activity and from week to week. We consider those sequences to be the courses' *designed* learning paths. But do learners actually adhere to these designed paths, or do they forge their own ways through the MOOCs? What are the implications of either following or not following the designed paths? Existing research has greatly emphasized, and succeeded in, automatically predicting MOOC learner success and learner dropout based on behavior patterns derived from MOOC learners' data traces. However, those predictions do not directly translate into practicable information for course designers & instructors aiming to improve engagement and retention — the two major issues plaguing today's MOOCs. In this work, we present a three-pronged approach to exploring MOOC data for novel learning path insights, thus enabling course instructors & designers to adapt a course's design based on empirical evidence.

Keywords

MOOCs, learning path analysis, visualization

1. INTRODUCTION

MOOCs can deliver a world-class education on virtually any academic or professional development topic to any person with access to the Internet. Millions of people around the globe have signed up to courses offered on platforms such as edX, Coursera, FutureLearn and Udacity. At the same time though, only a very small percentage of these learners actually complete a MOOC successfully [15], an issue that continues to plague massive open online learning. Keeping MOOC learners engaged and improving the dismal retention rates are major concerns to instructional designers and MOOC instructors alike. Considerable research efforts have been dedicated to the automatic prediction of learners' (imminent) dropout in MOOCs, e.g. [9, 12, 17, 24], under the assumption that once learners under the threat of attrition are identified, an automated intervention can be staged to (re)engage those learners with the course material. While the accuracy of these usually machine-learning-based predictors is high, their *explanatory power* is often low. Model features that have the strongest impact on prediction purely based on statistical grounds may not provide course designers & instructors with enough information to adapt the design or content of a MOOC in response.

In this work we aim to provide a more holistic view of learners' progression through a MOOC in order to enable more practicable insights to instructors and designers. Our approach to educational data mining as presented here is a very literal realization of Graesser's vision for the field by illustrating and "*look[ing] at unique learning trajectories of individuals*" [21]. We make use of the concept of *learning paths* (a learner's route through course activities) and investigate how the learning paths of successful and unsuccessful MOOC learners differ.

The design of MOOCs on the edX platform[1] implies a *linear* trajectory through the learning material. Most courses are broken up into weeks (Week 1, Week 2, etc.) and released one week at a time. Within these weeks, the standard instructional approach is to first provide a brief introduction to the week's material, followed by the weekly video lectures (the main source of content delivery), then the assessments that evaluate learners' knowledge of the preceding video lectures, and, finally, courses may offer bonus material. This cycle is repeated each course week (and sometimes multiple cycles comprise a single week). But do learners *actually* adhere to this cycle, and thus the *designed* learning path? Does it matter if they do not? These are the central issues that we focus on in this paper. While the concept of *executed* learning paths (i.e., the paths students actually take through a course) has received substantial attention in the e-learning and intelligent tutoring communities [13, 19], in the MOOC setting this concept has so far garnered little attention. First empirical evidence that learners do not always follow the designed sequence through a MOOC has been observed in [8], however, to our knowledge no in-depth investigation of this phenomenon in the MOOC context exists as of yet. We aim to close this knowledge gap and investigate the following

*The author's research is supported by the *Leiden-Delft-Erasmus Centre for Education and Learning.*
†The author's research is supported by the *Extension School* of the Delft University of Technology.

[1]Our empirical work is based on edX MOOCs, but the same principles apply to other major MOOC platforms.

research question:

*To what extent do learners **adhere** to a MOOC's designed learning path?*

We develop three approaches to characterize learning paths, thus providing three different views on a MOOC's *designed learning path* (created by the course instructor or designer) and the *executed paths* (created by the learners of the MOOC). We apply our approaches on the log traces of more than 113,000 learners who participated in one of four edX-based MOOCs in the domains of computer science, political debates and business ethics.
We show that (1) our approaches shed light on the deviations between designed and executed learning paths, and, (2) successful and unsuccessful learners differ considerably in the paths they follow. We believe that our work can provide instructional designers a valuable analysis tool to improve the design of both online courses and MOOC platforms in the future as they provide data-driven insights into the actual behavior of learners and the impact of their behaviors on learning outcomes.

2. RELATED WORK

In this section, we elaborate on existing research in learner modeling [5], focusing on works that investigate learning activity sequences and their impact on learning outcomes.
The problem solving behavior of learners in the context of e-learning and intelligent tutoring systems has been explored in [10, 13, 14, 19]. In contrast to our work, which considers a range of activities learners perform throughout a course (and compares them to the designed learning path), these works have explored learners' exhibited behavior within only one activity type — problem solving. Specifically, Köck and Paramythis [14] performed activity sequence clustering (an application of sequential pattern mining [22]) to model the learners' behavior, while in [13] automated clustering and human synthesis of the generated clusters were combined to identify patterns of problem solving. Shanabrook et al. [19] introduced a semi-automatic approach to identify a student's state while problem solving (including: gaming the system, guessing out of frustration, abusing hints, being on-task) in a high school-level intelligent tutoring system employing sequence-based motif discovery. Jeong and Biswas [10] developed a Hidden Markov Model to describe how different middle school student behavior trends lead to different learning processes & outcomes when problem solving.
In the context of MOOCs, sequences of learning activities have been explored by Wen and Rosé [23], who investigated the most common two-step activity sequences learners exhibit across two MOOCs. These patterns were then manually checked and analysed for interesting learning habits. A similar analysis of two-step chains was performed in Guo and Reinecke [8] who found that learners generally progress through the course content in a non-linear, "exploratory" manner [16]. Guo and Reinecke [8]'s observation of learners frequently performing "backjumps" (moving from a quiz to a lecture video previously introduced) can be considered as one of the first comparisons of executed and designed learning paths in MOOCs. Kizilcec et al. [11] (replicated in [6]) have also taken steps in this direction, by utilizing the assessment submission times (either on track, late or never) in MOOCs as indicators of learner engagement groups (completing, auditing, disengaging or sampling learners). Our work can be considered a significant expansion to these approaches, as we explore longer activity sequences (eight-step chains), thus enabling the discovery of more high-level and complex patterns and making designed vs. executed paths the focal point of our investigation.
Video interactions in MOOCs were the focus of Sinha et al. [20], who categorized the most prominent chains of video interactions (pause, play, speed, and skipping) and analyzed them with respect to learner dropout.MOOC discussion patterns have been investigated by Brooks et al. [3] who found that MOOC students exhibit markedly different discussion patterns than were expected based on blended learning environments. This finding can also be considered as a motivation for our work; MOOCs may not always be used by learners the way the instructors or course designers intended.
The concepts of process mining and conformance checking, in particular, are also employed in areas such as business process execution; [18] explains how business processes can be monitored (process mining) and then compared to the intended model (conformance checking) via a measure of fitness.

3. SUBJECTS & DATA

We explore our research question in the context of four MOOCs: `Functional Programming` (teaching the functional programming paradigm), `Data Analysis` (teaching spreadsheet and basic Python skills for data analysis), `Framing` (the art of political debates), and `Responsible Innovation` (a MOOC on the ethics and safety of new technologies). All MOOCs were offered on the edX platform in 2014/2015 and designed as xMOOCs.

Overview of MOOCs. Table 3 provides an overview of the four MOOCs in this study. The learner enrollment varies between ≈9k and ≈37k. While the four MOOCs are comparable in their video material offerings (between 41 and 59 videos), they differ significantly in the number of summative assessment questions (between 26 and 288 quiz questions). We also observe considerable differences in the percentage of video material watched by certificate-earning learners (replicating [8]) — less than half of the videos are accessed by successful learners in `Data Analysis`, while more than two thirds of the videos are accessed by successful learners in `Functional Programming`. Lastly, we note that the `Responsible Innovation` MOOC is an outlier with respect to the percentage of learners that passed the course *without* streaming any video material,[2] with nearly 20% of successful learners falling into this category; the same applies for only ≈4% of learners in the other three MOOCs.

Translating Log Traces into a Semantic Event Space. The edX platform provides a great deal of timestamped log traces, including clicks, views, quiz attempts, and forum interactions. We adapted the `MOOCdb`[3] toolkit to our needs and translated these low-level log traces into a data schema that is easily query-able.

For this work, we focus on four event types as listed in Table 2: events related to videos, quizzes, progress pages, and discussion forums. Videos can be watched - this event

[2]Note that the log traces did not capture video downloads and subsequent offline watching.
[3]`http://moocdb.csail.mit.edu/`

MOOC	Enrolled	Pass Rate	Chains Pass/Non-p.	Weeks	Videos	Quiz Questions	Passing Grade	Tries	Videos Accessed	Missing
Functional Programming	37,485	5.3%	1.06M/807k	14	41	288	60%	1	67.5%	4.3%
Responsible Innovation	8,850	4.3%	66k/30k	7	47	75	59%	1-3	49.7%	19.6%
Framing	34,017	2.4%	95k/141k	6	55	26	50%	2	51%	3.8%
Data Analysis	33,515	6.5%	1.02M/855k	8	59	136	60%	2	45%	3.6%

Table 1: Overview of the MOOCs in our study. The #Chains column contains the number of events observed throughout the MOOC (cf. Table 2). The "Passing Grade" shows the percentage of quiz questions to answer correctly to receive a course certificate. "Tries" indicates how many attempts a learner has per question. "Videos Accessed" shows the average % of course videos watched by certificate-earning learners. "Missing" is the % of certificate-earning learners who streamed zero video lectures.

Video	Quiz	Progress	Forum
WATCH	START	VIEW	START
	SUBMIT		SUBMIT
	END		END

Table 2: Overview of events considered in this work.

is generated whenever a user clicks the video 'play' button. Quizzes are identified through the beginning of the quiz session (the user enters the quiz page), the submission of one or more answers[4], and the ending of the quiz session (the user leaves the quiz page). Those quizzes are typically summative in nature. If a user views his or her progress page, the VIEW event is elicited. Finally, we condense discussion forum events into three kinds of items: the start of a forum session (the user first enters the forum), the submission of content (question, comment or reply) and the end of the forum session (the user leaves the forum page).

All *executed* learning paths that we extract from the learner log traces consist of the events listed in Table 2. The rationale for choosing these events comes from the designed learning path by which xMOOCs are typically formed: first watch one or more lecture videos, and then move on towards the quiz and/or forum section for assessment and knowledge building & verification respectively. In Figure 2 we visualize a week's designed learning path for each of the four MOOCs we study (this pattern is repeated in every course week). Video lectures form a common denominator, starting the path. Functional Programming and Data Analysis rely on videos and quizzes only (with Data Analysis exhibiting multiple video-quiz "cycles" within a week), whereas Responsible Innovation and Framing make use of the forums as well. The learning path shown for Framing does not include quizzes as they are posed only in the final week (in the form of an exam).

4. APPROACH

Having introduced the subjects of our work and the events we consider, we now describe the three distinct approaches to the visualization & exploration of executed learning paths (that is, learners' sequential movement *over time* through the activities offered in a MOOC) we developed.

[4]Note that on the edX platform answers to individual quiz questions are submitted (instead of all answers at once).

4.1 Video Interactions

As shown in Figure 2, videos are a focal point of xMOOCs. Accordingly, in a first analysis, we focus exclusively on video interactions and explore to what extent learners adhere to the designed video watching learning path. Therefore, in this study we only make use of WATCH events.

We transform the WATCH events generated by a set of learners L across the duration of a MOOC $\mathcal{M}$ into a directed graph $G_{\mathcal{M},L} = (V_{\mathcal{M}}, E_{\mathcal{M},L})$ — as the subscripts indicate, with $\mathcal{M}$ fixed, the set V is independent of the subset of learners chosen, while E is dependent on the learners in L. All lecture videos contained in $\mathcal{M}$ form the set of vertices $V_{\mathcal{M}}$. The vertices are labelled chronologically, that is, for any vertex pair (v_i, v_j) with $i < j$, the corresponding lecture video i must appear in the designed learning path before video j.

The edges are directed and weighted according to the number of WATCH events by the learners L: an edge between v_{i-1} (source) and v_i (target) presents the learners' transition between these videos, i.e. the number of times learners watching video v_{i-1} watch v_i next, before any other video. We disregard self-loops (watching the same video again) as we are focusing on the progression of the learners through the set of lecture videos.

Having generated $G_{\mathcal{M},L}$, we now turn to its visualization (to aid instructors and course designers): the vertex layout is sequential and governed by the designed learning path through the videos (represented as vertices). For MOOCs with thousands of participants it is likely that every single video pair combination possible is contained in at least one learning path. To avoid visual clutter, we filter out the most *infrequent* edges: we bin the edges according to the week their source vertex appears in and remove the 10% of edges that occur most infrequently in this course week.

To discover whether or not there are marked differences in the way different groups of learners behave, we generate the video interaction graph for different sets of learners, such as successful (certificate earning) vs. unsuccessful learners.

4.2 Behavior Pattern Chains

Having considered the transitions between lecture videos, we now turn to the exploration of transition patterns among all eight events identified in Table 2. Previous works [23] have viewed MOOC learner patterns either in terms of one-step directed pairs of events (such as *watch video* → *begin quiz*) or based on video click chains only [20].

One-step chains can only provide limited insights into more high-level behavioral patterns — we may, for instance, be in-

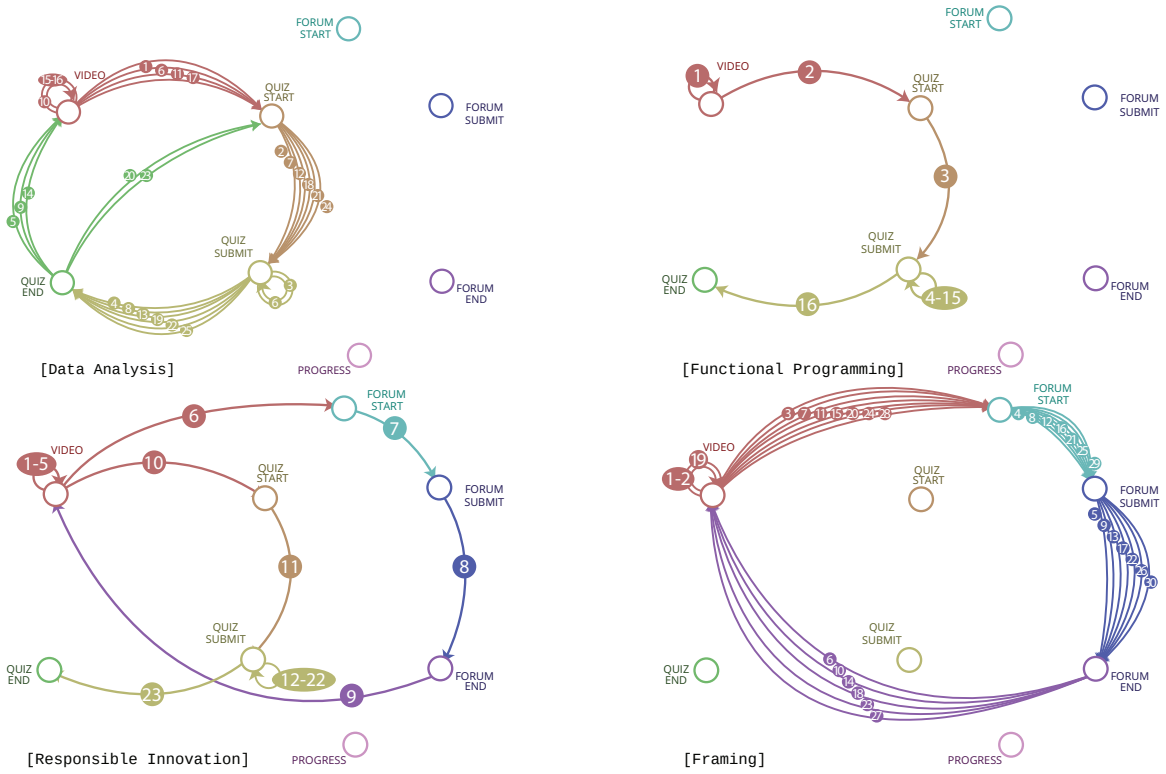


Figure 2: The designed learning path for a standard week (Week 4) of each MOOC. The circled numbers indicate the step number of each transition in that week's sequence. Notice the diversity in course designs that characterize these four MOOCs.

$$\texttt{QUIZ}_{START} \rightarrow \texttt{QUIZ}_{END} \rightarrow \texttt{WATCH} \rightarrow \texttt{WATCH}$$
$$\rightarrow \texttt{WATCH} \rightarrow \texttt{WATCH} \rightarrow \texttt{WATCH} \rightarrow \texttt{WATCH}$$

Figure 1: An example eight-step chain.

terested to understand how many learners are "binge watchers" (watching many videos in a row) or "strategic learners" (looking at quiz questions before watching the corresponding lecture video). In order to contribute insights to our research question we need to consider longer chains. We have settled on eight-step chains, as they provide insights into more high-level concepts but are still numerous enough in our log traces to make claims about their general usage. We consider all events of Table 2 and create event chains by sliding a window of size eight over each learner's chronologically ordered learning path through a MOOC. An example eight-step chain this procedure yields is shown in Figure 1.

To identify the underlying trends in the chains, we employed the *open card sort* approach [7]. After printing out two sets of the thirty most frequently occurring chains on paper, two authors independently sorted them into (non-predefined) like-groups by hand and afterwards discuss the differences in each sort, creating a composite of the two results. The outcome of this method is a synthesis of similar chain types into groups sharing the same *motif*, or recurring theme. Based on the motifs, we created a rule-based system that assigned a MOOC's entire set of chains to the identified motifs (chains that do not fit into any motif are left "unassigned"). This process is repeated for each of the MOOCs we investigate. The advantage of this approach over the automatic clustering of the chains is the infusion of our domain knowledge into the clustering process.

4.3 Event Type Transitions

Lastly, we explore event type transitions, or how likely learners are to move from one event type to another. Inspired by the methods employed in [10, 13, 14] we use discrete-time Markov chains (a memory-less state transitioning process encoding how often learners move from one event type to another) in order to chart the likelihood that a learner will transition from one engagement activity to another. Whereas the prior works employ these methods in the context of problem solving (knowledge assessment), we focus on the larger process of *knowledge building*, which transpires over the span of an entire course.

While it may be self-evident that non-passing learners answer less quiz questions than their certificate-earning peers (and thus the transition probabilities to $\texttt{SUBMIT}_{QUIZ}$ are likely to be lower for non-passers), the visualization of the Markov chains enables designers to pinpoint the differences in transitions between different types of learners (e.g. passers vs. non-passers) across all events in one coherent plot.

5. FINDINGS

To answer our research question (do learners adhere to the designed learning path?), we apply the three approaches outlined in Section 4 to the datasets described in Section 3.

5.1 Video Interactions

We visualize the video interactions across the first three weeks (these are where the most deviations occur; the later weeks are more in line with the designed path) of each MOOC in Figures 3 to 6, distinguishing two sets of learners: those that eventually earn a certificate ("Passing") and those that do not ("Non-Passing"). The *designed* video interaction learning path is exhibited by the left-to-right flow of the vertices (one per video). The edges correspond to the *executed* learning paths — with edge thickness indicating the (normalized) number of learners having taken that path (only the 90% most frequently occurring transitions each week are shown); the set of red edges represent the executed transitions that follow the designed transitions. A number of observations can be made based on the visualizations: (i) passing learners deviate considerably less from the designed learning path than non-passing learners across all four MOOCs, (ii) passing learners are more likely to skip video lectures introducing the platform (the first three videos in the `Framing` MOOC) than non-passing learners, indicating a higher level of seniority in MOOC-taking, (iii) towards the end of week three, the deviations among the sets of passing and non-passing learners are negligible (i.e. the non-passing learners still active exhibit a similar video watching behavior as the passers), and (iv) skipping videos — jumping ahead — is much more common than backtracking — jumping backwards — for both passers and non-passers.

An emerging object in the field of Design (and gaining some attention in the field of Software Design [4]) is that of *desire paths*, or paths not intended by the designer, but those which "arise due to off-[path] use ... for a variety of purposes such as access to places of interest and shortcutting" [2]. This research serves as a reminder that desire paths indeed exist in MOOCs (as evident in the skipping of introductory lecture material) — they just have not yet been made as visible as those brown stripes of beaten grass and dirt transecting public parks and trails. They are a reminder that humans can collectively communicate good design by their actions.

5.2 Behavior Pattern Chains

Our second approach explores learners' behavioral patterns. As outlined in Section 4.2, we first manually clustered and labelled the most frequent eight-step pattern chains in order to determine what type of behaviors (or motifs) learners exhibit beyond a single-click transition, before automatically assigning the remaining chains into those motifs. Depending on the MOOC, this approach yielded between eight and 11 motifs, with some motifs appearing only in a subset of courses. For brevity reasons, in Tables 3 to 6 for each MOOC we list its most frequent motifs (specifically those into which $\geq$2% of all chains are classified); as a comparison in Table 3 we also list the total number of chains generated by passing/non-passing learners in each MOOC — depending on the MOOC, the listed motifs capture between 42%–77% of the total number of chains. Whenever a motif is first introduced, we briefly describe which event types and event orderings characterize it[5].

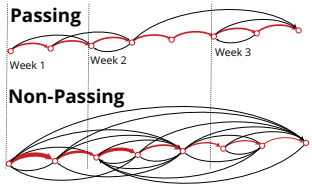


Figure 3: `Functional Programming` video interactions.

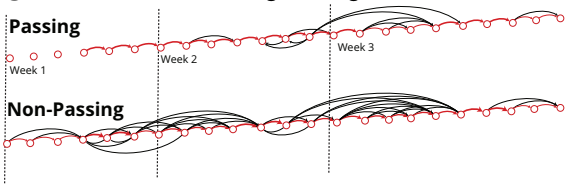


Figure 4: `Framing` video interactions.

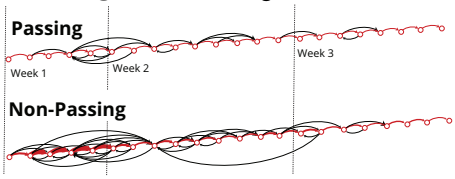


Figure 5: `Data Analysis` video interactions.

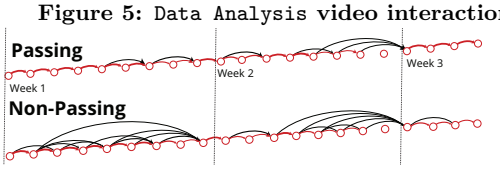


Figure 6: `Responsible Innovation` video interactions.

Examining the results, we observe that (i) *Binge Watching* is a frequent motif in all MOOCs with non-passers always exhibiting more binge watching (i.e. watching videos uninterrupted by other activities) than passers, (ii) the *Lecture→Quiz Complete* motif, which captures the "classic" xMOOC idea of video watching with subsequent question answering is frequent in three of the four MOOCs[6], however no consistent divergent behavior for passers and non-passers is found, (iii) motifs with forum events occur in three of the four MOOCs — by course design in `Framing` and `Responsible Innovation` (cf. Figure 2), but not in `Functional Programming`, indicating issues related to material clarity, and (iv) the *Quiz Check* motif, which is exhibited by learners checking the quiz questions without answering any of them (which is usually followed by video watching and subsequent quiz completion), is only found in one MOOC frequently; in `Data Analysis` 2% of the chains follow this motif, a smaller percentage than we expected, indicating that very few learners are gaming the system by "attempting to succeed in an educational environment by exploiting properties (quiz ques-

[5]Note, that we implemented our rules for the automatic assignment of chains to motifs according to these characterizations.

[6]It does not appear among the frequent motifs in `Framing`, which has a final exam instead of weekly quizzes.

tions are posted alongside the video material) of the system (edX platform) rather than by learning the material and trying to use that knowledge to answer correctly," [1].

	Motif	Freq. Total	Freq. Passing	Freq. Non-pass.
1	Quiz Complete	552,363 (29.4%)	328,995 (30.8%)	223,368 (27.7%)
	$\texttt{X}_{\texttt{QUIZ}}$ events only with at least one $X = \textit{SUBMIT}$			
2	Binge Watching	149,784 (8%)	59,498 (5.6%)	90,286 (11.2%)
	WATCH events only			
3	Lecture→Quiz Complete	100,179 (5.3%)	50,415 (4.7%)	49,764 (6.2%)
	WATCH event(s) followed by $\texttt{X}_{\texttt{QUIZ}}$ events; at least one $X = \textit{SUBMIT}$			
4	Quiz Complete→Forum	99,828 (5.3%)	67,722 (6.3%)	32,106 (4%)
	$\texttt{X}_{\texttt{QUIZ}}$ events (at least one $X = \textit{SUBMIT}$) followed by $\texttt{X}_{\texttt{FORUM}}$ events			
5	Quiz Complete→Progress	38,854 (2.1%)	26,126 (2.4%)	12,728 (1.6%)
	$\texttt{X}_{\texttt{QUIZ}}$ events (at least one $X = \textit{SUBMIT}$) followed by $\texttt{X}_{\texttt{Progress}}$ events			

Table 3: Most frequent motifs (≥2% chains) in Functional Programming.

	Motif	Freq. Total	Freq. Passing	Freq. Non-pass.
1	Quiz Complete	18,446 (16.6%)	11,377 (14.7%)	7,069 (21.1%)
2	Binge Watching	12,530 (11.3%)	8,461 (10.9%)	4,069 (12.1%)
3	Lecture→Quiz Complete	5,060 (4.6%)	3,752 (4.8%)	1,308 (3.9%)
4	Lecture→Forum→Lecture	3,910 (3.5%)	2,386 (3.1%)	1,524 (4.5%)
	WATCH events followed by $\texttt{X}_{\texttt{FORUM}}$ events followed WATCH events			
5	Quiz Complete→Progress	3,741 (3.4%)	2,898 (3.7%)	843 (2.5%)
6	Quiz Complete → Lecture → Quiz Complete	2,277 (2.1%)	2,019 (2.6%)	258 (0.8%)

Table 4: Most frequent motifs (≥2% chains) in Responsible Innovation.

5.3 Event Type Transitions

The Markov models of our four MOOCs are visualized in Figures 7 to 10. Since we observe the same event types across the four MOOCs, the set of vertices, their placement in the visualization, and their semantics are identical. To minimize visual clutter, we only plot the transitions (i.e. the edges) that exhibit a probability of 0.2 or higher. Once more we make the distinction between passing and non-passing learners. The resulting visualizations show the behavioral differences not only between passing and failing students within a given course, but these also allow for cross-course analyses which shed light on what types of behavioral patterns define a course. For example, when comparing **Framing** (Figure 9) and **Data Analysis** (Figure 7), marked differences in their pedagogical structure are evident; **Framing** appears to foster a very social, collaborative environment, whereas **Data Analysis** learners mostly focus their attention on lectures and assessments, with little concern for discussion. The visualizations also reveal at which specific moments learners seek feedback on their progress (i.e. make a transition to the Progress vertex), such as after a Quiz or Forum in **Responsible Innovation** and **Framing**. These movements are *not* included in any of the courses' designed paths; course designers can use this insight to proactively insert feedback in order to encourage more awareness and self-regulated learning. When comparing transitions of passing vs. non-passing learners, we observe that (i) non-passers make the transition to the video event from more diverse event types than passers (indicating that non-passers' executed paths follow the designed path to a lesser degree than passers' executed paths), (ii) video-to-video transitions are more prevalent among non-passers (in line with our findings on the binge watching motif), and (iii) passing learners are more likely to move from *Quiz Start* to *Quiz Submit*, while non-passing learners are more likely to move from *Quiz Start* to *Quiz End* (without answering a question).

	Motif	Freq. Total	Freq. Passing	Freq. Non-pass.
1	Binge Watching	64,822 (27.3%)	18,023 (18.9%)	46,726 (33.1%)
2	Lecture→Forum→Lecture	29,224 (12.3%)	11,651 (12.2%)	17,505 (12.4%)
3	Quiz Complete	12,984 (5.5%)	9,156 (9.6%)	3,781 (2.7%)
4	Forum→Lecture	7,850 (3.3%)	3,035 (3.2%)	4,800 (3.4%)
	$\texttt{X}_{\texttt{FORUM}}$ events followed WATCH events			
5	Lecture→Forum	7,488 (3.2%)	3,008 (3.2%)	4,462 (3.2%)
6	Quiz Complete→Lecture→Quiz Complete	5,551 (2.3%)	4,022 (4.2%)	1,501 (1.1%)

Table 5: Most frequent motifs (≥2% chains) in Framing.

	Motif	Freq. Total	Freq. Passing	Freq. Non-pass.
1	Quiz Complete	169,786 (9%)	116,878 (11.4%)	52,908 (6.2%)
2	Quiz Complete→Lecture→Quiz Complete	145,596 (7.7%)	82,247 (8%)	63,349 (7.4%)
3	Binge Watching	87,760 (4.7%)	28,066 (2.7%)	59,694 (7%)
4	Lecture→Quiz Complete	78,790 (4.2%)	41,543 (4.0%)	37,247 (4.4%)
5	Quiz Complete→Lecture	43,612 (2.3%)	21,916 (2.1%)	21,696 (2.5%)
6	Quiz Check	37,406 (2%)	19,444 (1.9%)	17,962 (2.1%)
	$\texttt{QUIZ}_{\textit{START}}$ followed by $\texttt{QUIZ}_{\textit{END}}$ events			

Table 6: Most frequent motifs (≥2% chains) in Data Analysis.

6. CONCLUSION

Before adaptive learning systems can reach their potential, two important baselines must be established: (i) the precise learning path the instructor wants the student to follow and (ii) students' natural behavior within the course. Adaptive instruction will be most effective when the differences between these two baselines are both identified and addressed. The present research offers novel insights into the identification of those differences.

Specifically, in this work we have introduced three different approaches (the video interaction graph, behavior pattern chains and event type transitions) to explore and visualize MOOC log traces with respect to the designed and executed learning paths.

We have applied our approaches on the log traces of four different edX-based MOOCs (from different domains and different pedagogical structures) and have shown to what extent learners (as a whole group as well as partitioned into passing and non-passing learners) follow the prescribed path. In future work, we will expand our analyses to a larger set of MOOCs to gain a greater understanding of the "classes" of xMOOCs that exist on the major MOOC platforms today. We also plan to consider more diverse sub-populations of learners in future analyses, beyond passing or not passing. We will also investigate semi-automatic approaches to the adaptation of MOOC learning paths, in order to minimize the gap between designed and executed paths as well as the impact this work has on engagement, retention, learner success and more fine-grained learner partitions (such as completing, auditing, and sampling learners [11]).

References

[1] Ryan S.J. Baker, Albert T. Corbett, Kenneth R. Koedinger, and Shelley Evenson *et al.* Adapting to when students game an intelligent tutoring system. In *Intelligent tutoring systems*, pages 392–401, 2006.

[2] Lori EA Bradford and Norman McIntyre. Off the beaten track: Messages as a means of reducing social trail use at St. Lawrence Islands National Park. *Journal of Park and Recreation Administration*, 25(1):1–21, 2007.

[3] Christopher Brooks, Jim Greer, and Carl Gutwin. The data-assisted approach to building intelligent technology-enhanced learning environments. In *Learning Analytics*, pages 123–156. 2014.

[4] Christian Crumlish and Erin Malone. *Designing social interfaces: Principles, patterns, and practices for improving the user experience*. O'Reilly Media, Inc., 2009.

[5] Michel C Desmarais and Ryan SJ d Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2012.

[6] Rebecca Ferguson and Doug Clow. Consistent Commitment: Patterns of Engagement across Time in Massive Open Online Courses (MOOCs). *Journal of Learning Analytics*, 2(3):55–80, 2016.

[7] Sally Fincher and Josh Tenenberg. Making sense of card sorting data. *Expert Systems*, 22(3):89–93, 2005.

[8] Philip J. Guo and Katharina Reinecke. Demographic differences in how students navigate through MOOCs. In *L@S 2014*, pages 21–30, 2014.

[9] Sherif Halawa, Daniel Greene, and John Mitchell. Dropout prediction in MOOCs using learner activity features. *Experiences and best practices in and around MOOCs*, 2014.

[10] Hogyeong Jeong and Gautam Biswas. Mining student behavior models in learning-by-teaching environments. In *EDM*, pages 127–136, 2008.

[11] René F Kizilcec, Chris Piech, and Emily Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *LAK 2013*, pages 170–179, 2013.

[12] Marius Kloft, Felix Stiehler, Zhilin Zheng, and Niels Pinkwart. Predicting MOOC dropout over weeks using machine learning methods. In *EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs*, pages 60–65, 2014.

[13] M. Köck and A. Paramythis. Towards Adaptive Learning Support on the Basis of Behavioural Patterns in Learning Activity Sequences. In *Intelligent Networking and INCOS '10*, pages 100–107, 2010.

[14] Mirjam Köck and Alexandros Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *User Modeling and User-Adapted Interaction*, 21(1-2):51–97, 2011.

[15] Daphne Koller, Andrew Ng, Chuong Do, and Zhenghao Chen. Retention and intention in massive open online courses. *Educause Review*, 48(3):62–63, 2013.

[16] Jens O Liegle and Thomas N Janicki. The effect of learning styles on the navigation needs of Web-based learners. *Computers in Human Behavior*, 22(5):885–898, 2006.

[17] Daniel FO Onah, Jane Sinclair, and Russell Boyatt. Dropout rates of massive open online courses: behavioural patterns. *EDULEARN14*, pages 5825–5834, 2014.

[18] Anne Rozinat and Wil MP van der Aalst. Conformance checking of processes based on monitoring real behavior. *Information Systems*, 33(1):64–95, 2008.

[19] David H Shanabrook, David G Cooper, Beverly Park Woolf, and Ivon Arroyo. Identifying high-level student behavior using sequence-based motif discovery. In *EDM*, pages 191–200, 2010.

[20] Tanmay Sinha, Patrick Jermann, Nan Li, and Pierre Dillenbourg. Your click decides your fate: Inferring Information Processing and Attrition Behavior from MOOC Video Clickstream Interactions. *EMNLP 2014*, pages 3–14, 2014.

[21] Sarah D. Sparks. Data Mining' Gains Traction in Education. *Education Week*, 2010.

[22] Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and Pang-Ning Tan. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. *SIGKDD Explor. Newsl.*, pages 12–23, 2000.

[23] Miaomiao Wen and Carolyn Penstein Rosé. Identifying latent study habits by mining learner behavior patterns in massive open online courses. In *CIKM '14*, pages 1983–1986, 2014.

[24] Diyi Yang, Tanmay Sinha, David Adamson, and Carolyn Penstein Rose. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *2013 NIPS Data-driven education workshop*, volume 11, 2013.

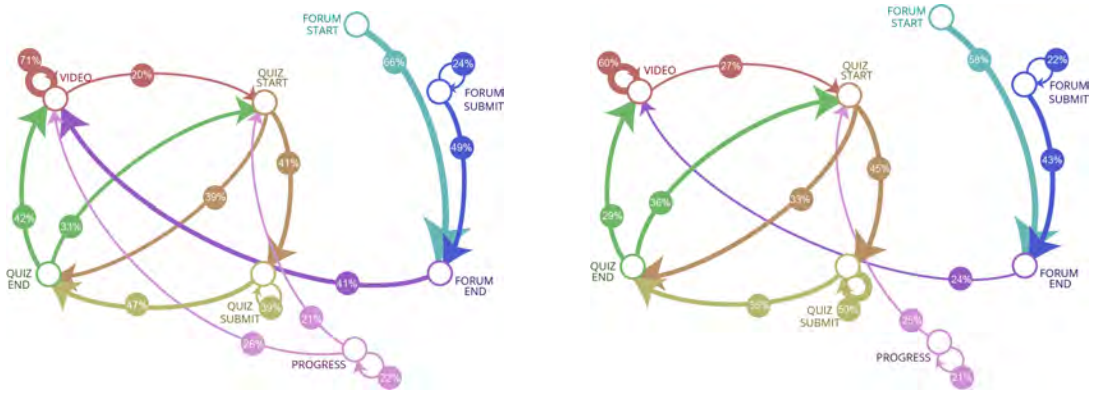


Figure 7: Markov Model state visualization of non-passing (left) and passing (right) learners in the `Data Analysis` MOOC. Edges with weights below 20% are hidden from view.

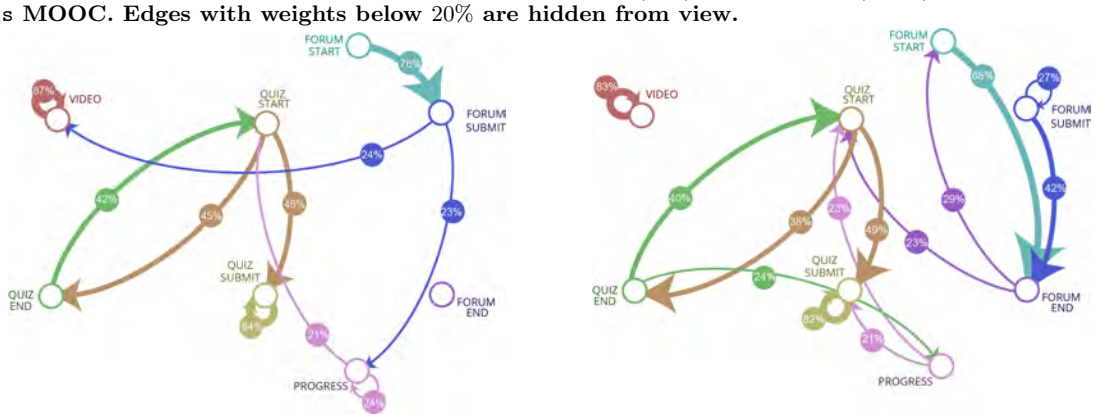


Figure 8: Markov Model state visualization of non-passing (left) and passing (right) learners in the `Functional Programming` MOOC. Edges with weights below 20% are hidden from view.

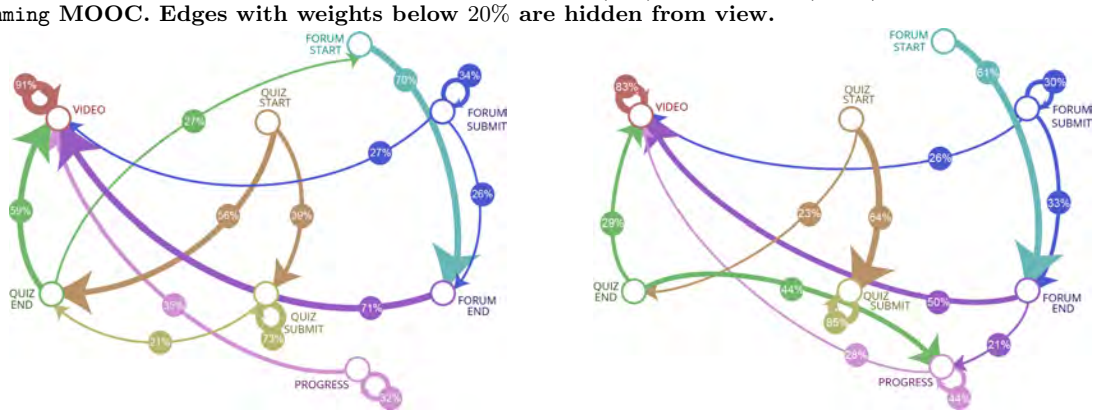


Figure 9: Markov Model state visualization of non-passing (left) and passing (right) learners in the `Framing` MOOC. Edges with weights below 20% are hidden from view.

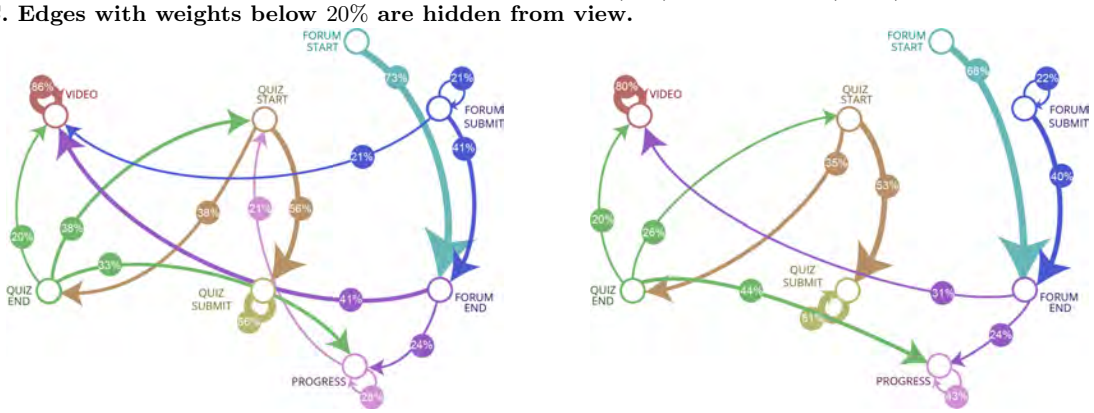


Figure 10: Markov Model state visualization of non-passing (left) and passing (right) learners in the `Responsible Innovation` MOOC. Edges with weights below 20% are hidden from view.

Dynamics of Peer Grading: An Empirical Study

Luca de Alfaro*
University of California, Santa Cruz
Department of Computer Science
luca@ucsc.edu

Michael Shavlovsky
University of California, Santa Cruz
Department of Computer Science
mshavlov@soe.ucsc.edu

ABSTRACT

Peer grading is widely used in MOOCs and in standard university settings. The quality of grades obtained via peer grading is essential for the educational process. In this work, we study the factors that influence errors in peer grading. We analyze 288 assignments with 25,633 submissions and 113,169 reviews conducted with CrowdGrader, a web based peer grading tool. First, we found that large grading errors are generally more closely correlated with hard-to-grade submission, rather than with imprecise students. Second, we detected a weak correlation between review accuracy and student proficiency, as measured by the quality of the student's own work. Third, we found little correlation between review accuracy and the time it took to perform the review, or how late in the review period the review was performed. Finally, we found a clear evidence of tit-for-tat behavior when students give feedback on the reviews they received. We conclude with remarks on how these data can lead to improvements in peer-grading tools.

1. INTRODUCTION

In peer grading, students review and grade each other's work. The grades assigned by the students to each item are then merged into a single *consensus grade* for the item. Peer grading has several benefits, as reported in the literature, including the fact that students learn from each other's work, and the reduced workload on the instructors. For these reasons, peer grading has been widely used both in MOOCs, where it would be infeasible for a small number of instructors to grade all work [14, 1, 5, 12], and in standard university classes [17, 15, 10, 18, 3, 16].

Successful peer grading is predicated on the ability to reconstruct a reasonably accurate consensus grade from the grades assigned by the students. This leads to the following question: what factors cause or influence the errors in peer-assigned grades? We are interested in this question for three reasons. First, we wish to obtain a better understanding of the dynamics and human factors in peer grading. Second, a better understanding of the causes of error has the potential to lead to tool improvements that reduce the errors. For example, if mis-understanding on the work submitted constituted a large source of error, then peer grading tools could be augmented with means for work authors and graders to communicate, so that the misunderstandings could be resolved. Third, a better model of peer grading errors might lead to better algorithms for aggregating the student-assigned grades into the consensus grades for each item.

Our interest in the origin of peer-grading errors is also due to our work on the peer-grading tool CrowdGrader [8]. We have put considerable effort in reducing the error in the consensus grade computed by CrowdGrader, as compared to control instructor-assigned grades. While efforts on the tool UI and UX paid off, as we will detail later, the efforts to create more precise grade-aggregation algorithms did not. In the context of MOOCs, [14] reports a 30% decrease in error using parameter-estimation algorithms that infer, and correct for, the imprecision and biases of individual users. CrowdGrader is used mostly in universities and high-schools. On CrowdGrader data, the parameter-estimation algorithm of [14] offers no benefit compared with the simple "Olympic average" obtained by removing lowest and highest grades, and averaging the rest. Indeed, we have spent a large amount of time experimenting with variations upon the algorithm (see also [7]) and new ideas, but we are yet to find an algorithm that offers consistent error reduction of more than 10% compared to the Olympic average. Thus our interest on the origin of errors in CrowdGrader: what are the main causes? What makes them so difficult to remove using algorithms based on parameter estimation, reputation systems, and more?

To gain an understanding of the dynamics of peer grading, we have analyzed a set of CrowdGrader data consisting in 288 assignments, 25,633 submissions, and 113,169 grades and reviews. Of the 25,633 submissions, 2,564 were graded by the instructors in addition to the students. The questions we ask include the following.

Is error mostly due to items or to students? We first ask the question of whether the imprecision in peer grades can be best explained in terms of students being imprecise, or items being difficult to grade. We answer this question in two different ways.

First, we build a parameterized probabilistic model of the review process, similar to the model of [14], in which every review error is the sum of a component due to the submission being reviewed, and of a component due to the reviewer. The parameters of the model are then estimated via Gibbs sampling [11]. The results indicate that students contribute roughly two thirds of the total evaluation error.

*In alphabetical order

This result, however, speaks to the *average* source of error. Of particular concern in peer grading are the very large errors that happen less frequently, but have more impact on the perceived fairness and effectiveness of peer grading. We measure the correlation of large errors in items, and in users; our results indicate that hard-to-grade items are a more common cause of large errors than very imprecise students.

Do better students make better graders? A natural question is whether better students make better graders. In Section 6 we give an affirmative answer: students whose submissions are in the lower 30%-percentile quality-wise have a grading error that is about 15% above average. The effect is fairly weak, a likely testament to the fundamental homogeneity in abilities in a high-school or college class, as well as to the fact that grading a homework is usually easier than solving the homework.

Does the timing of reviews affect their precision? In Section 7 we consider the relation of review timing and review precision. We did not detect strong dependencies between grading error and the time taken to complete a review, the order in which the student completed the reviews, or how late the reviews were completed with respect to the review deadline.

Does error vary with class topic? In Section 4 we consider the question of whether grading precision varies from topic to topic. Comparing broad topic areas, such as computer science, essays, science, we find the statistics to be quite similar, indicating how general factors are less important than the specifics of each class.

Does tit-for-tat affect review feedback? CrowdGrader allows students to leave feedback on the reviews and grades they receive; this feedback is then used as one of the factor that determines the student's grade in the assignment. The feedback was introduced to provide an incentive for writing helpful reviews. In Section 8 we show that when a grade is over 20% below the consensus, it receives a low feedback score due to tit-for-tat about 38% of the time.

In the next section, we give a brief description of CrowdGrader, and of the datasets on which our analysis is based. The subsequent sections present the details of the answers to the above questions. We conclude with a discussion on the nature of errors in peer grading, and on the implications for algorithms and reputation systems for computing consensus grades.

2. RELATED WORK

The accuracy of peer grading in the context of MOOCs has been analyzed in [13], where the match between instructor grade and student grades is analyzed in detail. The study finds a tendency by student to rate higher people that share their country of origin — and this in spite of the grading process being anonymous. The study finds that improvement in grading rubrics lead to improved grading accuracy. Geographical origin, along with gender, employment status, and other factors, are found to have influence on engagement in peer grading in a French MOOC in [4]. Our work is thus somewhat orthogonal to [4, 13]: we do not have data on student ethnicity, and we focus instead on factors measurable from the peer grading activity itself.

Frequently, peer grades are accompanied with reviewers' comments or feedback; [19] explores the possibility of using the review text to asses review quality. The authors show a successful application of classifiers and statistical Natural Language Processing to evaluate reviews.

Peer Instruction is a process in which students can observe grades by other reviewers, discuss the review, and consequently modify their grades [6]. The factors that influence grades in peer instruction have been studied in [2]. In spite of the different settings, [2] also observe that the behavior of high and low-scoring students is fairly similar in terms of their grading accuracy.

3. THE CROWDGRADER DATASET

To analyze the source of grading errors in peer grading, we rely on a dataset from CrowdGrader, a peer review and grading tool used in universities and high-schools [8]. After students submit their solutions to an assignment, students review and grade a certain number of submissions by their peers. From these peer grades, Crowdgrader computes a *consensus grade* for every submission. Once the review phase is concluded, the students can rate the reviews they received according to a 1 to 5-star rating. These review ratings are meant to provide an incentive for students to write detailed, helpful reviews of other students work.

The overall dataset we examined consisted in 288 assignments, for a total of 25,633 submissions and 113,169 reviews, written by 23,762 distinct reviewers. The number of reviewers is smaller than the number of submissions, as some students did not participate in the review phase. Table 1 gives a break-down of the dataset according to subject area. On average, each submission received 4.41 reviews, and each reviewer wrote on average 4.76 reviews.

We will refer to submissions also as *items*, and we will refer to students or reviewers also as *users,* thus adopting common terminology for general peer-review systems.

CrowdGrader includes three features that promote grading accuracy; these features likely influenced the data presented in this study.

Incentives for accuracy. The overall grade a student receives in a CrowdGrader assignment is a weighed average of the student's *submission, accuracy,* and *helpfulness* grades. The *accuracy grade* reflects the precision of the student's grade, compared either to the other grades for the same submission or, when available, to the instructor-assigned grade. The *helpfulness grade* grade reflects the rating received by the reviews written by the student. Combining the submission grade with the accuracy grade creates an incentive for students to be precise in their grading. The amount of incentive can be chosen by the instructor, but the default is to give 75% weight to the submission grade, 15% weight to the accuracy grade, and 10% weight to the helpfulness grade, and most instructors do not change this default.

Ability to decline reviews. Early in the development of CrowdGrader, we noticed that some of the most glaring grading errors occurred when reviewers were forced to enter a grade for submissions that they could not properly evaluate. This occurred, for instance, when students could not open the files uploaded as part of the submission, due to software incompatibilities. To mitigate this problem, we gave students the ability to *decline* to perform reviews of particular submissions. The total number of submissions a student can decline is bounded, to prevent students from "shopping around" for the easiest submissions to review.

Submission discussion forums. Another early source of large errors

	Assignments	Submissions	Reviewers	Reviews	Graded Assignments	Graded Submissions
Computer Science	188	19397	17829	86347	68	2402
Physics	7	274	270	907	6	33
Epidemiology	5	337	313	1551	0	0
Sociology	49	3822	3683	18339	3	16
Business	26	1217	1108	3915	15	106
English	9	397	383	1717	1	7
High-school	7	279	278	1097	5	20
Other	4	189	176	393	0	0
All Combined	288	25633	23762	113169	93	2564

Table 1: The CrowdGrader dataset used in this study. *Graded assignments* are the assignments where an instructor or teaching assistant graded at least a subset of the submissions. *Graded submissions* is the number of submissions that were graded by instructors or teaching assistants, in addition to peer grading.

in CrowdGrader consisted in gross mis-understandings between the author of a submission, and the reviewers. For instance, when zip archives are submitted, the reviewers may expect some information to be contained in one of the component files, whereas the author might have included it in another. Another example consists in mis-organizing the content of a software submission, so that the reviewers do not know how to run it and evaluate it. To remedy this, CrowdGrader introduced anonymous forums associated with each submission, where submission authors and reviewers can discuss any issues they encounter in evaluating the work.

4. ERRORS IN PEER GRADING

Instructor grades and Olympic averages. We measure review error as the difference between individual student grades, and the "consensus grade" for each submission. We consider two kinds of consensus grades. One is the *Olympic average* of the grades provided by the students: this is obtained by discarding the lowest and highest grade for each submission, and taking the average of the remaining grades. The other is the *instructor grade*. In CrowdGrader, instructors (or teaching assistants) have the option of re-grading submissions. In some assignments, instructors decided to grade most submissions as control; in other assignments, instructors mostly re-graded only submissions where student grades were in too much disagreement. When considering instructor grades, we consider only assignments of the first type, where instructors graded at least 30% of all submissions. Considering assignments where instructors grade only problematic submissions would considerably skew the statistics. The dataset, for instructor grades, is thus reduced to 19 assignments and 7675 reviews. Instructor and Olympic average grades have a coefficient of correlation $\rho = 0.81$ (with $p < 10^{-200}$), and an average absolute difference of 6.11 on the [0, 100] grading range.

Global and per-topic errors. Table 2 reports the size of errors in CrowdGrader peer grading assignments, split by assignment topic, and taking instructor grades and Olympic grades as reference. When the error is measured with respect to instructor grades, computer science, physics, and high-school assignments showed smaller average error than business, sociology and English, all of whose assignments required essay-writing. When the error is measured with respect to Olympic average, is is mainly business and English that show larger error.

5. ITEM VS. STUDENT ERROR

We consider in this section the question of whether error can be attributed predominantly to imprecise students, or to items that are difficult to grade.

	Average Error	N. of Assignments
Computer Science	7.52	15
Physics	10.6	1
Business	16.5	2
English	17.2	1
High School	10.6	1
All	7.67	19

(a) Error with respect to instructor grades, based on assignments with at least 30% of items graded by the instructor.

	Average Error	N. of Assignments
Computer Science	6.34	188
Physics	4.65	7
Epidemiology	4.57	5
Sociology	4.93	49
Business	7.7	26
English	8.37	9
High School	5.09	7
Other	8.15	4
All	6.16	288

(b) Error with respect to Olympic average.

Table 2: Mean absolute value difference error by topic. The grading range is normalized to [0, 100].

5.1 Average error behavior

To compare the contribution of students and items to grading errors, we develop a probabilistic model in which both students and items contribute to the evaluation error. The model is a modification of the $\mathbf{PG}_1$ model in [14], which allowed for student (but not item) error. In our model, each student has a *reliability* and each item has a *simplicity*; the variances of student and item errors are inversely proportional to their respective reliabilities and simplicities. Precisely:

$$\begin{aligned}
\text{(Reliability)}\ & \tau_u \sim \mathcal{G}(\alpha_0, \beta_0) \text{ for every student } u,\\
\text{(Simplicity)}\ & s_i \sim \mathcal{G}(\alpha_1, \beta_1) \text{ for every item } i,\\
\text{(True Grade)}\ & q_i \sim \mathcal{N}(\mu_0, 1/\gamma_0) \text{ for every item } i,\\
\text{(Observed Grade)}\ & g_{iu} \sim \mathcal{N}(q_i, 1/\tau_u + 1/s_i)\\
& \qquad \text{for every observed peer grade } g_{iu}
\end{aligned}$$

where $\mathcal{G}(\alpha, \beta)$ denotes the Gamma distribution with parameters α, β, and $\mathcal{N}(q, v)$ denotes the normal distribution with average q and variance v.

Given an assignment, we use Gibbs sampling [11] to infer the pa-

rameters $\alpha_0, \beta_0, \alpha_1, \beta_1, \mu_0, \gamma_0$. In order to apply Gibbs sampling, we need to start from suitable prior values for the quantities being estimated. To obtain suitable priors for the distribution of item quality, we first compute an estimated grade for each item using Olympic average, and we obtain μ_0 and γ_0 by fitting a normal distribution to the estimated grades. To estimate prior parameters α_0, β_0 of student reliabilities we fit a Gamma distribution to a set of approximated students reliabilities. In detail, for every student u we populate a list of errors l_u by the student. Again, we computer errors with respect to the average item grades after removing the extremes (the Olympic average). Using the list of error l_u, we estimate a standard deviation σ_u for every student $u \in U$. This allows us to approximate student reliability $\hat{\tau}_u$ as $\frac{1}{\sigma_u^2}$. Prior parameters α_0, β_0 are obtained by fitting a Gamma distribution to the set of estimated student reliabilities $\{\hat{\tau}_u | u \in U\}$. To estimate prior parameters α_1, β_1 for item simplicities we use the same approach as for α_0, β_0; the only difference is that item simplicities $\hat{s}_i$ are estimated using error lists l_i computed for every item i, rather than for every student u.

	students	items
Average Standard Deviation	14.2	6.4

Table 3: The average standard deviation of students and items errors computed over 288 assignment with 25633 items. The grading range is [0, 100].

Table 3 reports the average standard deviation of students and items inferred from the model. As we can see, students are responsible for over two thirds of the overall reviewing error.

5.2 Large error behavior

While students intuitively understand that small random errors will be averaged out, they are very concerned by large errors that, they fear, will skew their overall grade. Thus, we are interested in determining whether such large errors are more often due to students who are grossly imprecise, or items that are very hard to grade. In other words: do large errors cluster more around imprecise students, or around hard-to-grade items? We can answer this question because in CrowdGrader, items are assigned to students in a completely random way. Thus, any correlation between errors on items or students indicates causality.

We answer this question in two ways. First, we measured the information-theoretic *coefficient of constraint.* To compute it, let X and Y be two random variables, obtained by sampling uniformly at random two reviews x and y corresponding to the same item, or to the same student, and letting X (resp. Y) be 1 if x (resp. y) is incorrect by more than a pre-defined threshold (such as, 20% of the grading range for the assignment). Then, the mutual information $I(Y, X)$ indicates the amount of information shared by X and Y, and the coefficient of constraint $I(X, Y)/H(X)$, where $H(X)$ is the entropy of X, is an information-theoretic measure of the correlation between X and Y.

Tables 4 gives $I(X, Y)/H(X)$ for student and item errors, for different values of the error choice, and taking as reference truth for each item either the instructor grade, or the Olympic average for the item. When taking instructor grades as reference (Table 4a), large errors are about 5 times more correlated on items than on students, as measured by the coefficient of constraint. When Olympic grades are take as reference (Table 4b), large errors are about as correlated on items as they are on students. The difference in behavior is due

	Error Threshold				
	10%	15%	20%	25%	30%
Students	0.015	0.026	0.017	0.019	0.017
Items	0.075	0.082	0.082	0.1	0. 097

(a) Item errors computed with respect to instructor's grades. We use only assignments that have at least 30% of items grade by the instructor.

	Error Threshold				
	10%	15%	20%	25%	30%
Students	0.018	0.018	0.019	0.020	0.021
Items	0.045	0.030	0.020	0.021	0.020

(b) Item errors computed with respect to Olympic average.

Table 4: Coefficient of constraint $I(X, Y)/H(X)$ of large errors on the same item or by the same student, for different error thresholds.

to the fact that, when an instructor disagrees with the student-given grades on an item, this generates highly correlated errors on that item with respect to the instructor grade, but not with respect to the Olympic average. In any case, the results show that there is no particular correlation on students.

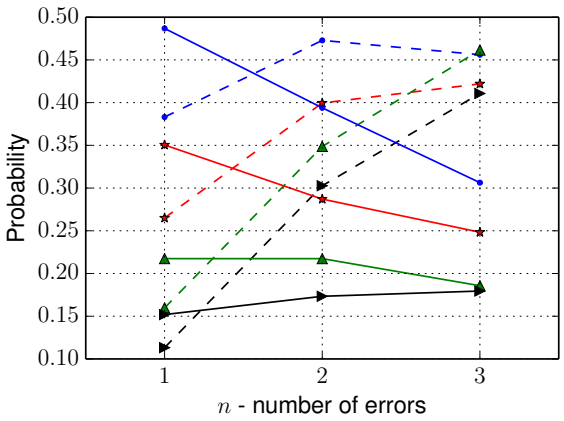
Another way to measure whether large errors tend to cluster around hard-to-evaluate items or around imprecise students consists in measuring the conditional probability $\rho_n = P(\xi \geq n | \xi \geq n-1)$ of an item (resp. student) having $\xi \geq n$ grossly erroneous reviews, given than it has at least $n-1$. If errors on an item (resp. reviewer) are uncorrelated, we would expect that $\rho_1 = \rho_2 = \rho_3 = \cdots$. If these conditional probabilities grow with n, so that $\rho_3 > \rho_2 > \rho_1$, this indicates that the more errors an item (resp. a student) has participated in, the more likely it is that there are additional errors. The values of $\rho_1, \rho_2, \rho_3, \ldots$ allow thus one to form an intuitive appreciation for how clustered around items or students the errors are.

The results are given in Figure 1. The data shows some clustering around users, for large errors of over 30% of the grading range. However, clustering around users seems weaker than clustering around items.

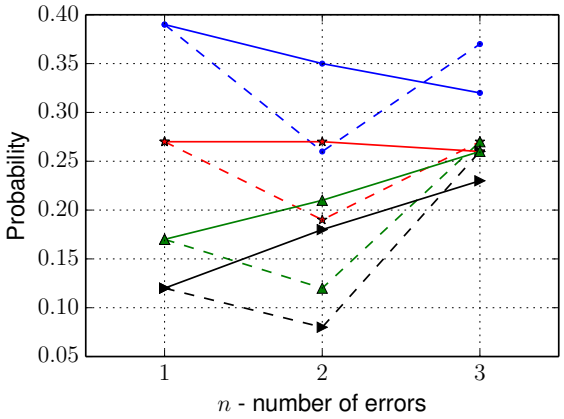
This provides a possible explanation for why reputation systems have not proved effective in dealing with errors in peer-graded assignments with CrowdGrader. Reputation systems are effective in characterizing the precision of each student, and taking it into account when computing each item's grade. Our results indicate however that errors in CrowdGrader are not strongly correlated with students, limiting the potential of reputation systems.

6. STUDENT ABILITY VS. ACCURACY

A natural question is whether better students make better graders. To answer this question, we can approximate the expertise of every student with the grade received by the student's own submission, and we can then study the correlation between the student's submission grade, and the review error. As we have only partial coverage of students with instructor grades, we compute the grade received by the student's own submission via Olympic average, rather than instructor grade. As the two generally are close, this increases coverage with minimal influence on the results. We study grading error with respect to both instructor grades and Olympic average.



(a) Errors computed with respect to the instructor's grades. We use only assignments that have at least 30% of items grade by the instructor.



(b) Errors computed with respect to Olympic average.

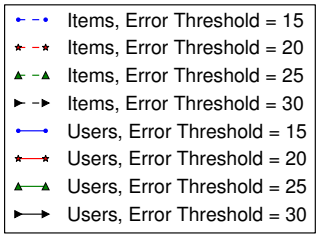


Figure 1: Conditional probabilities $\rho_n = P(\xi \geq n | \xi \geq n-1)$ of least n errors given at least $n-1$ errors. We considered error thresholds of 15%, 20%, 25%, 30%.

6.1 Aggregating data from multiple assignments

When aggregating data from multiple assignments, we cannot directly compare absolute values of grades, or absolute amount of time spent reviewing: each assignment has its own grade distribution, review time distribution, and so forth. To account for variation across assignments, we use the following approach. For each student there is an independent variable x, and an error e. In this section, x is the grade received by the student's own submission, measured via Olympic average; in the next section, x will be related to the time spent during the review, or the time at which the review is turned in. The error e is the difference, for each review, between the grade assigned as part of the review, and the grade of the reviewed submission, obtained either via Olympic average or via instructor grading.

First, for each assignment independently, we sort all students according to their x-value, and we assign them to one of 10 percentile bins: if the assignment comprises m students and the student ranks k-th, the student will be in the $\lceil 10k/m \rceil$ bin; we call these bins the 10%, 20%, ..., 100% bins. For each assignment a, we normalize the grading range to $[0, 100]$, and we let $n_{a,q}$ and $e_{a,q}$ be the number of students and the average error in the q percentile bin of assignment a, respectively. The average error for assignment a overall is thus $e_a = \sum_q n_{a,q} e_{a,q} / \sum_q n_{a,q}$. There are two ways of measuring the average error $e_{a,q}$ for one bin: as average absolute value error, or as average root-mean-square error. The two approaches lead to qualitatively similar conclusions, as we show later in this section. Due to lack of space, unless otherwise explicitly stated, we present here only the results for average absolute value, as they are somewhat less sensitive to rare large errors, and thus, more stable. The complete set of results is reported in [9].

We aggregate data from multiple assignments, computing for each percentile bin an absolute and a relative error, as follows. The *absolute* error e_q for each percentile q is computed as

$$e_q = \sum_a n_{a,q} e_{a,q} \Big/ \sum_a n_{a,q}. \quad (1)$$

The *relative* error r_q for each percentile q is computed as

$$r_q = \sum_a n_{a,q} \big(e_{a,q}/e_a\big) \Big/ \sum_a n_{a,q}, \quad (2)$$

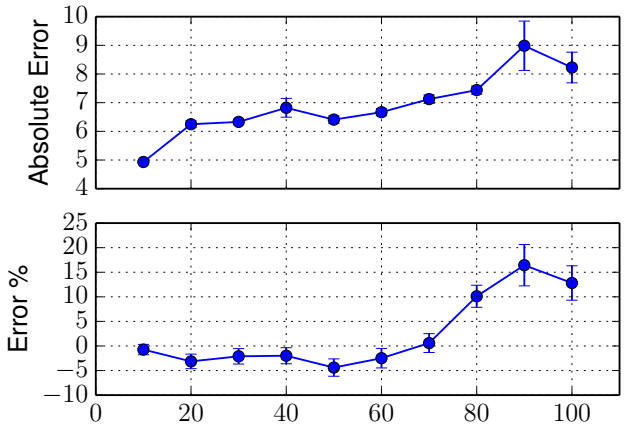
where $e_{a,q}/e_a$ is the relative error of bin q in assignment a.

6.2 Student ability vs. error

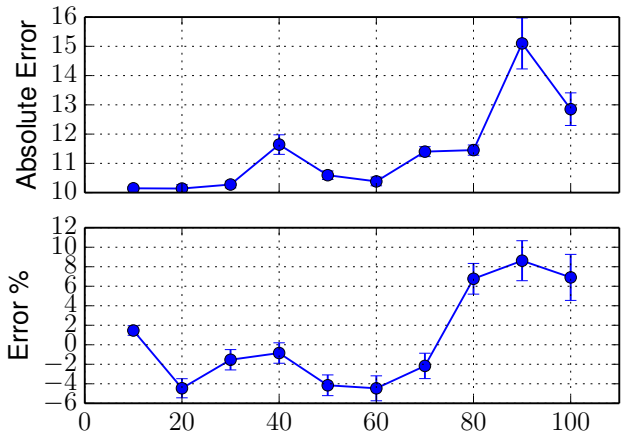
The data reported in Figure 2b shows the existence of some correlation between student submission grade, and grading precision, measured with respect to the Olympic average. In relative terms, students in the 80–100% percentile brackets show error that is 10% to 20% greater than students with higher submission grade. The absolute error tells a similar story. The two graphs do not have the same shape, due to the fact that relative errors are computed in (2) in a per-assignment fashion. In Figure 2a we report the same data, computed using rms error rather than average absolute value error. The data is qualitatively similar. Due to lack of space, in the remaining graphs we consider only average absolute error.

In Figure 3 we compare the error with respect to Olympic average with the error compared to instructor grades, for the subset of classes where at least 30% of submissions have been instructor-graded. While the absolute values are different, we see that the curves are very closely related, indicating that Olympic averages are a good proxy for instructor grades when studying relative changes in precision. The error with respect to instructor grades has very wide error bars for the 90% percentile, mainly due to the low number of data points we have for that percentile bracket in our dataset. We favor the comparison with the Olympic average, since the abundance of data makes the statistics more reliable.

The correlation between student ability (as measured by the submission score) and grading precision is lower than we expected. This might be a testament to the clarity of the rubrics and grading instructions provided by the instructors: apparently, such instructions ensure that most students are able to grade with reasonable precision the work by others. This may also be a consequence of the fundamental skill and background homogeneity of students in a classroom, as compared to a MOOC. We note that [2] also reported



(a) Mean absolute value difference error.



(b) Root mean square error.

Figure 2: Average grading errors arranged into authors' submissions quality percentiles. Grading errors and submission qualities are measured with respect to the Olympic average grades. The first percentile bin 10% corresponds to reviewers that have authored submissions with highest grades. Error bars correspond to one standard deviation.

low correlation between student grades and student precision in the related setting of peer instruction.

7. REVIEW TIMING VS. ACCURACY

We next studied the effect of the time taken to perform the reviews, and the order in which they were performed, on review accuracy. These measurements are made possible by the fact that CrowdGrader assigns reviews one at a time: a student is assigned the next submission to review only once the previous review is completed. This dynamic assignment ensures that all submissions receive a sufficient number of reviews. If each student were pre-assigned a certain set of submissions to review, as is customary in conference paper reviewing, then students who omitted or forgot to perform reviews could cause some submissions to receive insufficient reviews. CrowdGrader records the time at which each submission is assigned for review to a student, and the time when the review is completed. For these results, to conserve space, we provide the error only with respect to the Olympic average, for which we have more data. A comparison of error with respect to Olympic average

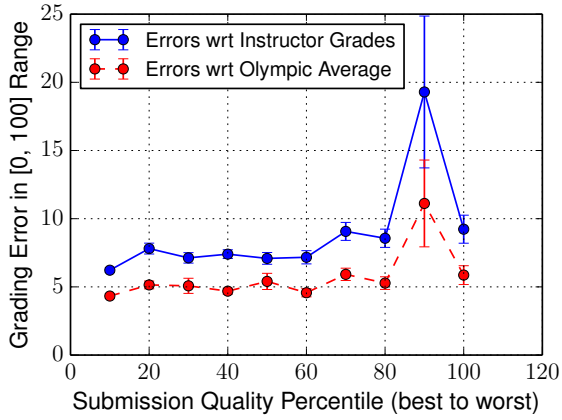


Figure 3: Average grading error arranged into authors' submission quality percentiles. The first percentile bin 10% corresponds to reviewers that have authored submissions with highest grades. We report the error both with respect to instructor grades, and to the Olympic average, considering only assignments for which at least 30% of submissions have been graded by instructors. Error bars correspond to one standard deviation.

and instructor grades confirms that the Olympic average is a good proxy for studying variation with respect to instructor grade also We omit the analogous of Figure 3 for the timing analysis due to lack of space; similarly, we include results only for mean absolute error. The complete result set is available in [9].

Time to complete a review. We first considered the correlation between the time spent by students performing each review, and the accuracy of the review; the results are reported in Figure 4. The results indicate that reviews that are performed moderately quickly tend to be slightly more precise. The correlation is weaker than we expected. We expected to find error peaks due to students that spent very little time reviewing, and that entered a quick guess for the submission grade, rather than performing a proper review. There are no such peaks: either students are very good at quickly estimating submission quality, or they mostly take reviewing and seriously in CrowdGrader. We believe the latter hypothesis is likely the correct one: for instance, in many computer science assignments, there is no good way of "eye-balling" the quality of a submission without compiling and running it.

Time at which a review is completed. Next, we studied the correlation between the absolute time when reviews are performed, and the precision of the reviews. Figure 5 shows the existence of a modest correlation: the reviews that are completed in the first 10% percentile tend to be 10% more accurate than later reviews. The effect is rather small, however. In a typical CrowdGrader assignment, students are given ample time to complete their reviews, and the reviews themselves take only one hour or so to complete. Students likely do not feel they are under strong time pressure to complete the reviews, and time to deadline has little effect on accuracy.

Order in which reviews are completed. Lastly, we study whether the order in which a student performs the reviews affects the accuracy of the reviews. We are interested in the question of whether students learn while doing reviews, and become more precise, or whether they grow tired and impatient as they perform the reviews, and their accuracy decreases. Figure 6 shows that the accuracy of

students does not vary significantly as the students progress in their review work. Evidently, the typical review load is sufficiently light that students do not suffer from decreased attention while completing the reviews.

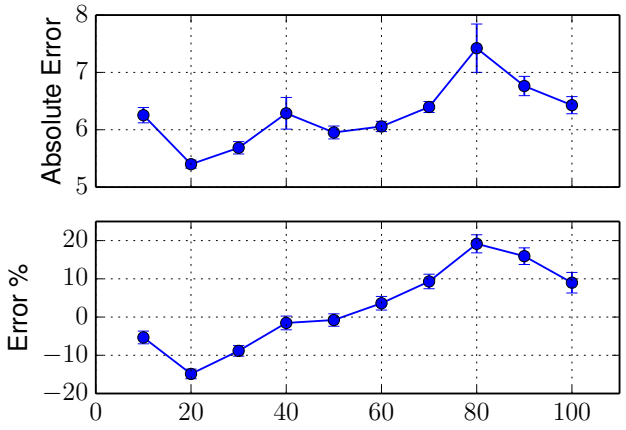


Figure 4: Absolute and relative grading error vs. the time employed to perform a review; the first percentile bin 10% corresponds to reviews with shortest review time. The grading range is normalized to $[0, 100]$, and the error is measured with respect to the Olympic average. The error bars indicate one standard deviation.

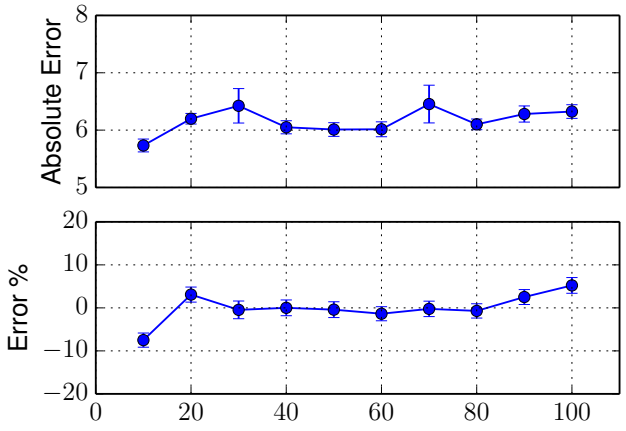


Figure 5: Absolute and relative grading error vs. absolute time when a review is completed. The first percentile bin 10% corresponds to the 10% of reviews that were completed first among all assignment reviews. The grading range is normalized to $[0, 100]$, and the error is measured with respect to the Olympic average. The error bars indicate one standard deviation.

8. TIT-FOR-TAT IN REVIEW FEEDBACK

In CrowdGrader, students can leave feedback to each review and grade they receive. The feedback is expressed via 1-to-5 star rating systems as follows:

- 1 star: factually wrong; bogus.
- 2 stars: unhelpful.
- 3 stars: neutral.
- 4 stars: somewhat helpful.
- 5 stars: very helpful.

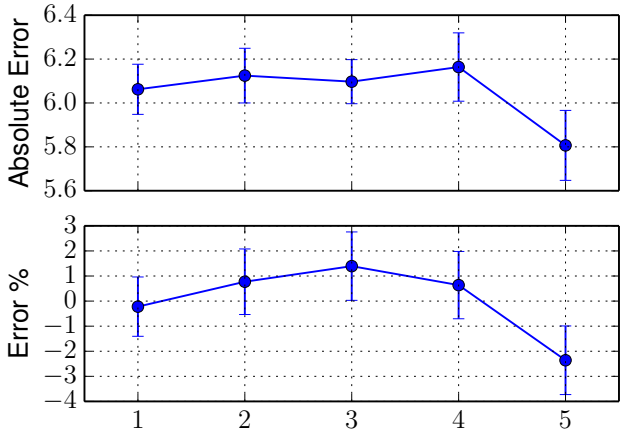


Figure 6: Absolute and relative grading error vs. ordinal number of a review by a student. The review 1 is the first a student performs, 2 is the second, and so forth. The grading range is normalized to $[0, 100]$, and the error is measured with respect to the Olympic average. Error bars indicate one standard deviation.

Many such ratings are given as tit-for-tat: when a student receives a low grade, the student responds by assigning a low feedback score (typically, 1 star) to the corresponding review. Indeed, CrowdGrader includes a technique for identifying such tit-for-tat, so that students, whose overall grade depends also on the helpfulness of their reviews, are not unduly penalized. We were interested in analyzing the question of how prevalent tit-for-tat is.

Overall, review grade and review feedback have a correlation of 0.39, with a p-value smaller than 10^{-300}. The correlation between grade and feedback indicates tit-for-tat, as there is no reason why lower grades should per-se be associated with written reviews that are less helpful. Interestingly, the correlation is fairly independent from the subject area. To bring the tit-for-tat into sharper evidence, we computed also the following statistics. We consider a grade a p (resp. n) outlier if the grade is over 20% above (resp. below) the Olympic average. We then measured the conditional probabilities P_p, P_n that p and n outliers would receive a one or two-star rating, conditioned over the probability that the reviews received a rating at all (students do not always rate the reviews they receive). Over all assignments, we measured $P_p = 0.06$ and $P_n = 0.44$. Since there is no a-priori reason why overly negative reviews may be of worse quality than overly positive ones, the excess probability $P_n - P_p = 0.38$ can be explained by tit-for-tat. This shows that tit-for-tat is rather common: for grades that are 20% or more below the consensus, there is a 38% probability of low feedback due to tit for tat. Fortunately, it is easy to discard low ratings given in response to below-average grades, as CrowdGrader does.

9. DISCUSSION

We presented an analysis of a large body of peer-grading data, gathered on assignments that used CrowdGrader across a wide set of subjects, from engineering to business and humanities. Our main interest consisted in identifying the factors that influence grading errors, so that we could devise methods to control or compensate for such factors. Out results can be thus summarized:

- Large errors are no more strongly correlated on students than

they are on items. In other words, students who are imprecise on many submissions are not a dominant source of error.

- There is some correlation between the quality of a student's own submission (which is an indication of the student's accomplishment), and the grading accuracy of the student, but the correlation is weak and limited to the student with highest, and lowest submission grades.
- There is little correlation between the accuracy of a review, and the time it took to perform the review, or how late in the review period the review was performed.
- There is clear evidence of tit-for-tat behavior when students give feedback on the reviews they receive.

All of the correlations we measured, except for the tit-for-tat one, are rather weak. This is a reassuring confirmation that peer-grading works as intended.There are no large sources of uncontrolled error due to factors such as student fatigue in doing the reviews, or gross inability of weaker students to perform the reviews. The peer-grading tool, in our classroom settings, ensures that the remaining errors are fairly randomly distributed, with little remaining structure.

The results highlight the difficulties in using reputation systems to compute submission grades in peer-grading assignments in high-school and university settings. Reputation systems characterize the behavior of each student, in terms for instance of their grading accuracy and bias, and compensate for each student's behavior when aggregating the individual review grades into a consensus grade. However, our results indicate that the large errors that most affect the fairness perception of peer grading are most closely associated with items, rather than with students. Reputation systems are powerless with respect to errors caused by hard-to-grade items: even if they can correctly pinpoint which submissions are hard to grade, little can be done except flagging them for instructor grading. Indeed, the reputation system approach of [14], which yielded error reductions of about 30% for MOOCs, yielded virtually no benefit in our classroom settings.

There is more potential, instead, in approaches that make it easier to grade difficult submissions. In CrowdGrader, we introduced anonymous forums, associated with each submission, where submissions authors and reviewers can discuss any issues that arise while reviewing the submission. These forums are routinely used, for instance, to solve the glitches that often arise when trying to compile or run code written by someone else. Anectodally, these forums have markedly increased the satisfaction with the peer-grading tool, as students feel that they have a safety net if they make small mistakes in formatting or submitting their work, and are in the loop should any issues occur.

10. ACKNOWLEDGEMENTS

This research has been supported in part by the NSF Award 1432690.

11. REFERENCES

[1] S. P. Balfour. Assessing Writing in MOOCs: Automated Essay Scoring and Calibrated Peer Review (TM). *Research & Practice in Assessment*, 8, 2013.

[2] S. Bhatnagar, M. Desmarais, C. Whittaker, N. Lasry, M. Dugdale, and E. S. Charles. An analysis of peer-submitted and peer-reviewed answer rationales, in an asynchronous peer instruction based learning environment. *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[3] D. Chinn. Peer assessment in the algorithms course. In *ACM SIGCSE Bulletin*, volume 37, pages 69–73. ACM, 2005.

[4] M. Cisel, R. Bachelet, and E. Bruillard. Peer assessment in the first french mooc: Analyzing assessors' behavior. *Proceedings of the 7th International Conference on Educational Data Mining*, 2014.

[5] S. Cooper and M. Sahami. Reflections on Stanford's MOOCs. *Communications of the ACM*, 56(2):28–30, 2013.

[6] C. H. Crouch and E. Mazur. Peer instruction: Ten years of experience and results. *American Journal of Physics*, 69(9):970–977, 2001.

[7] L. de Alfaro and M. Shavlovsky. CrowdGrader: Crowdsourcing the evaluation of homework assignments. *Technical Report UCSC-SOE-13-11, UC Santa Cruz, arXiv:1308.5273*, 2013.

[8] L. de Alfaro and M. Shavlovsky. CrowdGrader: A tool for crowdsourcing the evaluation of homework assignments. In *Proceedings of the 45th ACM technical symposium on Computer science education*, pages 415–420. ACM, 2014.

[9] L. de Alfaro and M. Shavlovsky. Dynamics of peer grading: An empirical study. Technical Report UCSC-SOE-16-04, School of Engineering, University of California, Santa Cruz, 2016.

[10] E. F. Gehringer. Electronic peer review and peer grading in computer-science courses. *ACM SIGCSE Bulletin*, 33(1):139–143, 2001.

[11] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–741, 1984.

[12] K. F. Hew and W. S. Cheung. Students' and instructors' use of massive open online courses (MOOCs): Motivations and challenges. *Educational Research Review*, 12:45–58, 2014.

[13] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, and S. R. Klemmer. Peer and self assessment in massive online classes. In *Design Thinking Research*, pages 131–168. Springer, 2015.

[14] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller. Tuned models of peer assessment in moocs. *arXiv preprint arXiv:1307.2579*, 2013.

[15] R. Robinson. Calibrated Peer Review: an application to increase student reading & writing skills. *The American Biology Teacher*, 63(7):474–480, 2001.

[16] J. Sadauskas, D. Tinapple, L. Olson, and R. Atkinson. CritViz: A Network Peer Critique Structure for Large Classrooms. In *EdMedia: World Conference on Educational Media and Technology*, volume 2013, pages 1437–1445, 2013.

[17] K. Topping. Peer assessment between students in colleges and universities. *Review of educational Research*, 68(3):249–276, 1998.

[18] A. Venables and R. Summit. Enhancing scientific essay writing using peer assessment. *Innovations in Education and Teaching International*, 40(3):281–290, 2003.

[19] W. Xiong, D. J. Litman, and C. D. Schunn. Assessing reviewer's performance based on mining problem localization in peer-review data. In *EDM*, pages 211–220. ERIC, 2010.

Sequence Matters, But How Exactly? A Method for Evaluating Activity Sequences from Data

Shayan Doroudi[1], Kenneth Holstein[2], Vincent Aleven[2], Emma Brunskill[1]
[1]Computer Science Department, [2]Human-Computer Interaction Institute
Carnegie Mellon University
{shayand, kjholste, aleven, ebrun}@cs.cmu.edu

ABSTRACT

How should a wide variety of educational activities be sequenced to maximize student learning? Although some experimental studies have addressed this question, educational data mining methods may be able to evaluate a wider range of possibilities and better handle many simultaneous sequencing constraints. We introduce Sequencing Constraint Violation Analysis (SCOVA): a general method for evaluating alternative activity sequences using existing data. SCOVA can be used to explore many complex sequencing constraints, such as prerequisite relationships, blocking, interleaving, and spiraling. We demonstrate SCOVA on data collected from a fractions intelligent tutoring system (ITS). Some of our findings challenge our initial hypotheses regarding sequencing, illustrating the utility and versatility of the method. The method can also be applied to other learning environments, as long as the available data has substantial variability in students' activity sequences.

1. INTRODUCTION

How does the sequencing of pedagogical activities impact student learning? Answers to this question can both contribute to core learning sciences knowledge, as well as have important practical implications for how educational activities should be sequenced in order to maximize learning. As such, there has been significant interest in this issue, and prior research suggests that student learning can be quite sensitive to temporal sequencing (e.g., [16, 1, 15, 17]).

Prior work that tackles this problem mostly falls into either theoretical analyses or empirical studies. Unfortunately, conducting theoretical analyses of the cognitive demands of individual tasks and the interdependencies among multiple tasks [7, 10, 3] can be prohibitively time consuming for large curricula. In addition, such analyses may be particularly vulnerable to various cognitive biases, such as expert blind spots [12]. Considerable experimental research has examined the effects of activity sequencing along various dimensions, including interleaving versus blocking of topics [1, 17] and sequencing of activities according to the degree of scaffolding they provide [15, 8]. However, such classroom experimental studies typically compare only two or three possible conditions, in contrast to the enormous number of orderings possible (at least exponential in the number of activity categories of interest).

An educational data mining approach could allow us to evaluate a much broader range of possible orderings in order to better understand which sequences may be optimal. Moreover, it might be possible to apply such techniques to any datasets that have considerable variation in how they order instructional content for students. These include datasets generated from educational technologies that present activities in a partially or fully randomized order (e.g., [13]), those that adaptively present activities in response to measured student variables (e.g., [4]), and those that provide students with some degree of control over activity selection (e.g., [11]).

We are particularly interested in investigating which orderings over a variety of topics and activity types are most effective for maximizing student learning and performance. Prior educational data mining approaches have focused on examining pairwise dependencies between instructional items (e.g., individual skills, problems, or problem sets) in a curriculum, in order to infer underlying prerequisite structure [5, 21, 18]. The prerequisite structures learned via such methods could be used, for example, to inform adaptive problem selection algorithms that avoid presenting a given item until the student is believed to have mastered its prerequisites [7]. Other methods for detecting ordering effects over instructional items have additionally relied upon the use of fitted Bayesian Knowledge Tracing (BKT) models [13, 19], and have thus depended upon strong assumptions about student learning. Whereas these prior approaches are typically limited to discovering pairwise relationships between items, and have tended to assume that these items are presented in a blocked fashion, we wish to examine the impacts on student learning and performance of more complex (and potentially softer) sequencing constraints.

We investigate the question of optimal topic and activity type sequencing in the context of our fractions intelligent tutoring system (ITS) [6]. Our tutor covers three broad topics (making and naming fractions, fraction equivalence and ordering, and fraction addition) and three different types of activities that correspond to learning mech-

anisms in the theoretical Knowledge-Learning-Instruction (KLI) framework: sense-making, induction and refinement, and fluency-building processes [9]. While previous experimental work has investigated the optimal sequencing of activity types under the KLI framework [14], there has been little empirical work investigating the optimal sequencing of topics in a fractions curriculum, and no work to our knowledge examining how the optimal sequencing of activity types may vary across topics.

We develop a general-purpose method for leveraging log data to evaluate and compare different ways of sequencing activities. We believe our method for evaluating sequencing constraints can be utilized to discover how to sequence activities in a variety of learning environments. We tested our method on log data from our fractions tutor and found results that countered our initial hypotheses on how to order both topics and activity types. We also found that the optimal ordering over KLI activity types may vary from topic to topic, but that for the most part, these orderings were consistent with what was suggested by prior literature [14].

2. SEQUENCING CONSTRAINT VIOLATION ANALYSIS (SCOVA)

We first describe our general method, and then present the particular instantiations of our method that we used in our analyses in Section 3. Sequencing Constraint Violation Analysis (SCOVA) is a method for analyzing different sequencing constraints and identifying which ones lead to the best student performance. SCOVA takes as input a set of student trajectories (which contains the sequence of problems given to each student and the students' responses to those problems) and a cost function for each set of sequencing constraints that one wants to evaluate. The cost function is a function over student trajectories that specifies how often a particular set of sequencing constraints is violated; in particular, it assigns to each student's sequence a number of violations up to the total length of the sequence.

Many different types of sequencing constraints can be considered. For example, one sequencing constraint could be that a student must be given at least one instance of problem type X before the student is given problem type Y. For this constraint, whenever problem Y is presented to a student before any instance of problem X, that student trajectory incurs one violation. Another constraint could be that problem X should *always* appear immediately before problem Y, so whenever a student sees problem Y without seeing problem X right before it, that sequence incurs a violation. For such constraints, the cost function is simply the number of problems where the constraint is violated. However, another sequencing constraint could suggest that a student's trajectory should match a particular desirable sequence, and our cost function in that case could be the Levenshtein distance[1] between the student's sequence and the desirable sequence. We can also consider sets of more than one sequencing constraints: for example, the constraints could specify that problem X should come before problem Y and problem Y should come before problem X. In this case, the cost function counts every time *any* constraint is violated.

[1]The Levenshtein distance, often referred to as edit distance, is a standard measure of distance between two sequences, measuring the smallest number of insertions, deletions, and substitutions to change one sequence into another. It is a valid cost function since it takes on a value between 0 and the length of the sequences.

Unlike many existing methods (e.g., [13, 21, 19]), SCOVA is not limited to evaluating pairwise orderings. Indeed, SCOVA can handle much more general constraints on order sequencing, such as blocking, interleaving, and spiraling. SCOVA can also handle constraints that depend not just on the prior history of problems given, but also on the student's performance and interactions (such as performance on prior activities, pretest score, or measures of affect).

Given the cost functions and student trajectories, SCOVA proceeds as follows for each set of sequencing constraints that we want to evaluate. We first use the cost function to compute the proportion of violations for every student's sequence by dividing the cost of the sequence by the length of the sequence. We next use the proportion of violations as an input variable in a linear regression model that predicts some measure of student performance (e.g., within-tutor performance, posttest score, or learning gains), and fit the parameters that maximize the log likelihood of the resulting model.

To evaluate the impact of a particular set of sequencing constraints, we look at two measures. First, we compute the Bayesian Information Criterion (BIC) of the linear regression model fit for violations of those constraints. This provides us with a way to compare different sequencing constraints; a model with a lower BIC score provides a better fit of the student data (as evaluated by log likelihood, adjusted for the number of parameters of the model). However, BIC alone simply measures predictive fit, not whether the sequencing constraints are beneficial for students or harmful. To understand whether the sequencing constraints may have a positive or negative impact on the outcome variable, we look at the sign of the coefficient of the violation variable in the fit linear model. We limit our attention to models where the proportion of violations has a negative coefficient—that is, models where violating the sequencing constraints is associated with worse student performance. Among these models, we can then compare the sequencing constraints by comparing the BICs of their models.

Recall that SCOVA can handle multiple sequencing constraints conjunctively (e.g., example problem X should come before Y and Y before Z). This makes the most sense when the different sequencing constraints are mutually exclusive, i.e., we cannot incur more than one violation on any particular problem. However, we may want to consider different sequencing constraints that can occur simultaneously and perhaps constrain different aspects of student trajectories (e.g., for example one might constrain the ordering of topics and the other might constrain the ordering of activity types). SCOVA can be extended to simultaneously consider the impact of these different sequencing constraints *disjunctively*. To do so, we learn a predictive linear regression model with one input variable for each set of sequencing constraints. When we have more than one set of sequencing constraints in our model, we focus our attention on models that have negative coefficients for *every* predictor corresponding to violations of sequencing constraints. If the BIC of a model

that takes two sequencing constraints into account is lower than that of each of the models that consider just one of the sequencing constraints individually, it suggests that both ordering constraints are important but capture different aspects of student performance. We can also compare the relative effects of violating different sequencing constraints by comparing the coefficients within the same model.

3. EVALUATION DOMAIN

As a concrete example, we now describe how we used our proposed approach to evaluate the impact of ordering on student learning and performance when using an online fractions tutor for fourth and fifth grade fractions topics [6]. The tutor covers topics emphasized in the Common Core, a set of non-binding national standards for mathematics education in the US: making and naming fractions on the number line (MN), fraction equivalence and ordering (EQ), and fraction addition (ADD).[2] The tutor was originally developed to investigate the potential benefits of using a broader range of instructional activity types than is typical of an ITS. Tutor activities were designed to promote each of the 3 categories of learning mechanisms posited under the KLI framework [9]: sense-making (SM), induction and refinement (IR), and fluency-building (F). The tutor's curriculum includes activities targeting each of these categories of learning mechanisms, for each of the main topics.

Under KLI, SM processes correspond to "explicit, verbally mediated learning in which students attempt to understand or reason" [9], IR processes are defined as non-verbal learning processes that improve the accuracy of knowledge, and fluency processes are non-verbal processes that strengthen memory and enable students to apply their procedural knowledge faster and more fluently. As such, SM activities in our tutor were designed to promote conceptual understanding through an interleaving of brief instructional videos with exercises intended to support self-explanation. By contrast, IR activities in our fractions tutor were designed to emphasize procedural learning and practice via fine-grained task decomposition and step-level guidance – as is typical of ITSs [20]. Finally, fluency-building activities were designed to promote the development of fluent performance on minimally decomposed problem-solving exercises. A more detailed description of our operationalization of these three activity types can be found in [6].

3.1 Sequencing Constraints

We consider a variety of sequencing constraints over both topics and activity types in our analyses. Since we have three topics and three activity types there are six potential orderings of each. For each of the following constraints (aside from the baselines at the end) we consider them with respect to each of the six possible orderings (for either topics or activity types).

[2]In the fractions tutor, activities within each of these three broad topics broke down further into multiple subtopics. For example, fraction equivalence and ordering activities included activities on finding common denominators, reducing fractions, and identifying equivalent fractions using number lines, among other subtopics. In addition, individual activities typically targeted a number of finer-grained skills.

3.1.1 Exposure-Based Constraints

Exposure-based constraints stipulate that students be exposed to (i.e., carry out) one topic/activity type a certain number of times before being exposed to the next. Every time the student receives a problem before being exposed to its "prerequisite" enough times, a violation is incurred. We define two categories: *Exposure-based topic constraints* require that students do at least one problem of a topic before seeing a problem of the next topic. *Exposure-based type constraints* require that within each topic, students should do one problem of an activity type before seeing the next activity type, without constraining the order of topics. Note that we can have the ordering over activity types fixed for every topic, or we can let it vary. If we let it vary, there are $6^3 = 216$ possible exposure-based *varying* type constraints.

3.1.2 Performance-Based Constraints

Performance-based constraints stipulate that students should reach a certain level of within-tutor performance on a topic/activity type before being exposed to the next. Every time the student receives a problem when their recent performance on its "prerequisite" is not beyond some threshold, a violation is incurred. Notice that even though such a constraint may be satisfied for a given student at a certain point in time, it is possible that it will no longer be satisfied later on, if the student's performance drops. *Performance-based topic constraints* require that students' performance on the last 10 steps of the topic should be beyond some topic-specific threshold before they receive problems for the next topic. (These steps may be from one problem or span over several problems.) By contrast, *performance-based type constraints* require that within each topic, students' performance on the last 10 steps on a particular activity type should be beyond some threshold specific to that topic-type pair before they receive problems of the next activity type (for the given topic). As before, in addition to the six type constraints that are fixed per topic, we have 216 possible performance-based *varying* type constraints.

We selected thresholds to detect a basic level of competency with problems of a particular activity type within a topic—a lower bar than mastery. The thresholds shown in Table 1 were obtained by taking the average student performance on the last 10 steps upon doing two problems of the given topic or topic-type pair

3.1.3 Blocking and Interleaving-N Constraints

To show the flexibility of the SCOVA method in considering sequencing constraints beyond straightforward prerequisite relationships, we consider whether topics and activity types should be interleaved or blocked with respect to topics/types. We measure violations in terms of Levenshtein distance from a particular sequence. The *blocking topic constraint* stipulates that for every student, the first third of their sequence (rounding up) should correspond to the first topic, the second third (rounding up) should correspond to the second topic, and the last third should correspond to the last topic. This is not a sequence we would typically be able to assign in practice, because we do not generally know how many problems a student will do ahead of time, but it represents a pure form of blocking while guaranteeing students see all of the activity types. The *interleaving-N topic constraints*, for $N = 1, \ldots, 6$, require sequences that

MN	EQ	ADD	MN/SM	MN/IR	MN/F	EQ/SM	EQ/IR	EQ/F	ADD/SM	ADD/IR	ADD/F
0.453	0.360	0.206	0.415	0.514	0.125	0.356	0.547	0.308	0.262	0.158	0.269

Table 1: Thresholds used for performance-based topic and type constraints. Notice that for the type constraints, we have distinct thresholds for each topic. The thresholds were obtained by taking the average student performance on the last 10 steps upon doing two problems of the given topic or topic-type pair.

give N problems of the first topic followed by N problems of the second topic followed by N problems of the third topic. However, if a student did less than $3N$ problems in total, we instead use the sequence used for the blocking constraint, in order to check whether they get reasonable exposure to all three topics.

3.1.4 Proportion-Only Baselines

To see if ordering topics or activity types actually matters, we compare to baselines that just use the proportions of topics or activity types in the sequence as predictors to predict within-tutor performance. Note that our two baselines each have two predictors (e.g., for activity types, we have one for proportion of SM and proportion of IR; the proportion of fluency-building activities is linearly dependent on the first two and so it is not needed in the model).

3.2 Hypotheses

We started data analysis with several hypotheses about the best order of topics and activity types. We note however that in order to illustrate our method, the specific hypothesized best order does not matter, although it does matter in illustrating that the method can produce unexpected (but reasonable) results.

3.2.1 Topic Dependencies

Our first hypothesis is that in early fractions learning, topics build on each other in the following way. MN helps students build a basic representation of fractions as numbers that have a magnitude, represented by their place on the number line. This representation is hypothesized to help in building an understanding of the notion of equivalence and the notion that fractions can be compared and ordered in terms of their magnitude. Moreover, equivalence would appear to be a strict prerequisite for addition of fractions with unlike denominators, because fractions with unlike denominators need to be converted to equivalent fractions before they can be added. Thus, the hypothesized best topic order is MN-EQ-ADD. Topics may not need to be fully blocked (i.e., presenting all MN activities before any EQ activities, and all EQ activities before any ADD activities), but it may be better for students to initially be exposed to topics in this order and perhaps continue to see the different topics in an interleaved fashion (as interleaving has been show to be beneficial [1, 17]).

3.2.2 Type Dependencies

As mentioned, the KLI framework distinguishes between three distinct classes of learning mechanisms, SM, IR, and F. It does not, however, make any claims regarding the order in which these processes might be most effective or even whether each class of mechanisms is needed when learning in a complex domain (such as fractions). There has been little prior work investigating how instructional activities targeting each of the KLI activity types can best be sequenced to maximize student learning and performance. However, [14] previously found that presenting students with SM activities before presenting them with fluency-building activities is beneficial when teaching connection making between multiple graphical representations of fractions. Given the dearth of prior work in this area, we do not have very strong expectations regarding the best order of these different activity types within a topic. However, in line with the work by [14], our hypothesis is that SM-targeting activities should come first, then IR-targeting activities, and finally, F-targeting activities. A second reason to expect that it is effective to do IR activities before F activities is that in our tutors, IR activities provide more elaborate scaffolding than F activities. As before, we do not mean to suggest a fully blocked ordering may be best, but also consider orders that interleave activity types with the hypothesized SM-IR-F order strictly observed early on.

3.3 Data

We collected data from 347 students using our ITS (in 20 classrooms across four different schools). The data was initially collected for a randomized control trial comparing three adaptive problem selection policies and two non-adaptive policies. The three adaptive policies had quite a bit of variation in the kinds of trajectories given to students; they thus provide data that is a good fit for SCOVA. However, the non-adaptive policies resulted in trajectories that were identical in how they sequenced topics and activity types, so we did not use data from those policies in our analyses (leaving 211 students). Students were given a pretest, followed by using the tutor for typically four class periods, and were finally given a posttest that was identical to the pretest. Each student worked at their own pace and completed as many problems as they could during the allotted time, resulting in a tail of students who did many more problems than average. This could present a confound in our analysis since students who do many problems are more likely to be high performing students, as well as violating sequencing constraints less than others (because they are likely to do many problems after satisfying all sequencing constraints). We thus limited our analyses to students who did 60 or fewer problems (197 students).

3.4 Modeling

In the SCOVA framework, we fit a linear regression model with predictors corresponding to the proportion of violations of one or more sets of sequencing constraints. The outcome variable we used was the within-tutor performance of students on all problems of the tutor with each topic-type pair having an equal weight (e.g., each student's performance on MN/SM problems has an equal weight to their performance on EQ/F problems). If a student received no problems of a

	Topic Constraints			Type Constraints	
	Exposure	Performance		Exposure	Performance
MN-EQ-ADD	**-236.28**	**-299.69**	SM-IR-F	**-226.16**	**-236.84**
EQ-MN-ADD	**-244.39**	**-319.13**	IR-SM-F	**-208.59**	**-218.94**
MN-ADD-EQ	**-201.04**	**-274.17**	SM-F-IR	-193.39	**-200.89**
EQ-ADD-MN	**-201.26**	**-254.75**	IR-F-SM	**-196.85**	**-217.20**
ADD-MN-EQ	**-193.81**	**-199.80**	F-SM-IR	**-202.91**	**-224.57**
ADD-EQ-MN	-205.73	**-193.84**	F-IR-SM	**-192.97**	**-200.32**
Proportion-Only	-233.48		Proportion-Only	-201.77	

Table 2: Comparison of BICs of individual exposure-based and performance-based constraints as well as proportion-only baselines. Aside from the proportion-only baselines, BICs corresponding to models where the coefficient of the predictor is negative are shown in bold. The smallest BIC in each column is underlined.

	SM-IR-F	IR-SM-F	SM-F-IR	IR-F-SM	F-SM-IR	F-IR-SM
MN-EQ-ADD	**-246.09**	**-232.81**	-232.95	**-231.28**	**-231.11**	-234.97
EQ-MN-ADD	**-249.30**	**-251.63**	-242.24	**-247.12**	**-240.24**	**-239.11**
MN-ADD-EQ	**-224.69**	**-208.31**	-197.71	**-198.37**	**-202.08**	-196.00
EQ-ADD-MN	**-223.54**	**-217.35**	-201.94	**-203.99**	**-200.60**	-197.16
ADD-MN-EQ	**-225.26**	-205.57	-188.94	-191.63	**-197.83**	-188.64
ADD-EQ-MN	-227.92	-219.54	-200.48	-208.98	-210.61	-201.07

Table 3: Comparison of BICs of models combining exposure-based topic and type constraints. BICs corresponding to models where the coefficients of both predictors are negative are shown in bold. The smallest BIC is underlined.

topic-type pair, then the average is only over the topic-type pairs they received. One could also add other predictors to improve the model fits and potentially control for other confounds. We add the student's pretest score as a predictor to all of our models as this improved the model fit.

4. RESULTS

Table 2 shows the BICs of models with only a single ordering constraint predictor corresponding to performance-based and exposure-based topic and type sequencing constraints in addition to BICs of the two proportion-based baselines. First, we notice that the lowest BIC models using exposure-based and performance-based ordering constraints have a better fit than the baseline models, which, as mentioned, only consider the proportion of activities given for either topic or activity type. This suggests that ordering of topics and activity types makes a difference beyond just the frequency with which they appear.

Second, we find that the lowest BICs for the sequencing constraints over topics are lower than the lowest BICs for sequencing constraints over activity types, especially for the performance-based constraints. This suggests that sequencing over topics might be more important than activity type ordering. This is also supported by the coefficients in the fitted linear regression models; for example, the coefficient for the best fitting performance-based topic constraints is -0.37, whereas for the best fitting performance-based type constraints, it is -0.23.

Third, for both the exposure-based and the performance-based constraints, the models for EQ-MN-ADD have the lowest BICs among all the topics models and the models for SM-IR-F have the lowest BICs among all the types models. We also find that the models that put fractions addition first either have the worst BICs or have positive coefficients (i.e., violation of constraints correlates with increased student performance), which makes sense, as we really do not think students should be doing addition (potentially with unlike denominators) before fraction equivalence. Likewise, the models with the best BICs and largest negative coefficients are the ones that put ADD last.

Finally, we find that the performance-based constraints have lower BICs than the exposure-based constraints. This reasonably seems to suggest that students' within-tutor performance can be predicted more accurately when we take into account the extent to which individual students reached a basic level of competence on one topic/type before being exposed to the next topic/type. We must note, however, that for the performance-based metric, the number of violations is impacted by a student's performance, and is thus related to the outcome variable in a confounded way. For example, a student who does very well on the tutor would be more likely to get fewer performance-based violations for any sequence than a student who does poorly on the tutor, partially explaining the lower BICs for performance-based models than exposure-based models. While we cannot conclude that performance-based constraints are better than exposure-based constraints from this analysis, we hypothesize that the relative ranking of different orders of topics/types may not be impacted severely by this confound.

To start to understand the interaction of type and topic ordering constraints on within-tutor student performance, we fit linear regression models that used two prerequisite violation input variables: one for one of the six topic orderings, and one for one of the six type orderings. Table 3 shows

	SM-IR-F	IR-SM-F	SM-F-IR	IR-F-SM	F-SM-IR	F-IR-SM
MN-EQ-ADD	**-319.39**	**-297.22**	**-301.80**	**-298.25**	**-299.90**	**-296.39**
EQ-MN-ADD	**-328.84**	**-330.35**	**-314.33**	**<u>-336.70</u>**	**-330.46**	**-317.38**
MN-ADD-EQ	**-300.02**	**-285.10**	**-270.36**	**-283.20**	**-286.98**	**-269.96**
EQ-ADD-MN	**-269.67**	**-280.47**	**-249.80**	**-279.55**	**-261.14**	**-250.23**
ADD-MN-EQ	**-239.09**	**-215.61**	**-203.15**	**-214.25**	**-220.07**	**-199.02**
ADD-EQ-MN	**-233.34**	**-213.69**	**-196.73**	-211.92	**-219.29**	**-195.28**

Table 4: Comparison of BICs of models combining performance-based topic and type constraints. BICs corresponding to models where the coefficients of both predictors are negative are shown in bold. The smallest BIC is underlined.

	Exposure-Based		Performance-Based	
	Coefficient	p-value	Coefficient	p-value
Intercept	0.37	$< 2 * 10^{-16}$	0.45	$< 2 * 10^{-16}$
Pretest	0.025	$3.45 * 10^{-8}$	0.023	$4.77 * 10^{-10}$
Topic Violations	-0.20	$8.07 * 10^{-7}$	-0.36	$< 2 * 10^{-16}$
Type Violations	-0.17	$4.20 * 10^{-6}$	-0.22	$2.27 * 10^{-10}$
BIC	-260.77		-355.00	
Adjusted r^2	0.39		0.62	

Table 5: Best fitting models incorporating both topic constraints and varying type constraints. The lowest BIC model according to exposure-based constraints suggests IR-SM-F for EQ, SM-IR-F for MN, and F-IR-SM for ADD, and the lowest BIC model according to performance-based constraints suggests the ordering IR-SM-F for EQ, SM-IR-F for MN, and IR-SM-F for ADD.

the BICs for all 36 models that have pairs of violations of exposure-based topic and type constraints as predictors, and Table 4 shows analogous results for pairs of performance-based constraints. We find that both for exposure-based and performance-based constraints, the model with the lowest BIC uses the EQ-MN-ADD ordering over topics, but for exposure-based constraints the ordering over activity types is IR-SM-F, while for performance-based constraints it is IR-F-SM. Note that this is different from the lowest BIC ordering of activity types when using only type constraints (SM-IR-F, see Table 2). However, we find that for many other orderings over topics (e.g., MN-EQ-ADD and MN-ADD-EQ), the model with the lowest BIC is the one with the SM-IR-F ordering over activity types. This suggests that the best ordering over activity types may depend on how we sequence the topics.

Indeed, the best ordering over activity types might vary from topic to topic (e.g., to maximize student performance it may be best to give IR first for EQ but SM first for MN). To test this possibility, we searched for the lowest BIC model with a predictor corresponding to some *varying* type constraints and a predictor for one of the six topic constraints[3]. The lowest BIC model according to exposure-based constraints suggests the ordering IR-SM-F for EQ, SM-IR-F for MN, and F-IR-SM for ADD (although several models were within three BIC points including ones that suggests IR-SM-F for ADD), and the lowest BIC model according to performance-based constraints suggests the ordering IR-SM-F for EQ, SM-IR-F for MN, and IR-SM-F for ADD (although, again, several models were within three BIC points including ones that suggests IR-F-SM for ADD). Table 5 shows the coefficients and fits for both of these lowest BIC models. Notice that the coefficients for the topic constraints have larger magnitudes than those for the varying type constraints (although not much larger in the exposure-based model), suggesting again that sequencing over topics is more important than sequencing over activity types. Moreover, the coefficients of the topic and activity type constraints violation variables in Table 5 are not only highly significant (i.e., significantly different than 0), but also their magnitudes are quite substantial given the outcome variable is bounded between 0 and 1. This suggests that students who receive activities in an order that has a large proportion of sequencing constraint violations would be expected to have considerably worse performance on the tutor problems.

Finally, we turn to models based on blocking and interleaving constraints. Table 6 shows the results comparing interleaving-N constraints and blocking constraints for all six orderings over topics. Again we find that the model corresponding to the EQ-MN-ADD order has the lowest BIC, but interleaved in chunks of four problems. This agrees with our hypothesis that one should not simply present the topics in a blocked fashion. Interestingly, most of the other models, including ones corresponding to fully interleaving or blocking, have equally bad BICs, regardless of the topic order.

5. DISCUSSION

Our novel method for evaluating activity sequences led to a number of interesting findings about sequencing topics and activity types in our tutor, illustrating the utility of the method. We found that all of the models fit using various topic sequencing constraints unanimously suggested that

[3]This results in 1296 models to search over, as there are $6^3 = 216$ different varying type constraints and six different topic constraint orderings

	Interleaving-1	Interleaving-2	Interleaving-3	Interleaving-4	Interleaving-5	Interleaving-6	Blocking
MN-EQ-ADD	**-193.09**	**-201.03**	**-198.85**	**-198.33**	**-197.80**	**-195.52**	**-195.63**
EQ-MN-ADD	**-195.89**	**-197.01**	**-198.89**	**-211.94**	**-202.93**	**-194.93**	**-193.91**
MN-ADD-EQ	**-194.04**	**-193.27**	**-195.00**	-194.00	**-193.01**	**-195.47**	**-193.01**
EQ-ADD-MN	**-194.75**	**-193.81**	**-194.06**	**-196.07**	**-194.03**	**-193.08**	-194.08
ADD-MN-EQ	**-193.49**	**-193.14**	-192.97	-194.11	-194.62	-193.34	-193.50
ADD-EQ-MN	**-193.62**	**-193.04**	**-192.96**	-196.66	-203.76	-197.92	-195.97

Table 6: Comparison of BICs of models with interleaving-N constraints and blocking constraints. BICs corresponding to models where the coefficient of the predictor is negative are shown in bold.

EQ-MN-ADD is the best way to sequence topics (suggesting that students should at least have some exposure to EQ before MN and some exposure to MN before ADD). This challenges our initial hypothesis that MN-EQ-ADD is the optimal ordering for learning. This result seems to indicate that, in contrast to our hypothesis, learning to make and name fractions (MN) on the number line may be facilitated by knowledge and skill regarding fraction equivalence and ordering (EQ), more so than the other way around. This result may suggest that an understanding of relationships between multiple fractions can help with learning about making and naming individual fractions on the number line, to a greater degree than previously realized. However, we cannot rule out alternative explanations. For example, it could be that our tutor activities are not successful in helping students learn knowledge that transfers to other topics. We note that in the MN activities, students used the number line extensively, whereas they did not in the EQ activities; in the latter they almost exclusively used the symbolic notation of fractions. It may be that if both topics had used the number line, the work on making and naming fractions might have facilitated learning about equivalence and ordering more. Thus, our method for evaluating sequences raises questions about tutor design, which, if and when resolved, could potentially lead to a more effective tutor.

The results on sequencing of activity types were not as unequivocal. We found that the best sequence over activity types may well vary for topics, which is itself an interesting result. For MN and EQ, the models suggest SM should precede F. This result agrees with prior literature on how to order sense-making and fluency activities [14]. However, the relative ordering of SM and IR is not as clear, with it possibly being advantageous to give IR activities before SM activities in many cases, challenging our initial hypothesis.

One may wonder if our results can simply be explained in terms of ordering topics and activity types from easiest to hardest. However, this does not seem to be the case. Note that the performance thresholds in Table 1 provide a measure of difficulty for each topic and each topic-type pair. Based on this measure of difficulty, MN would be classified as easier as EQ, but we saw that our models suggest EQ should come before MN. Furthermore, according to this measure of difficulty, ADD/IR problems would be classified as the most difficult for fraction addition; however, our lowest BIC types models suggest that IR should either come first or second for fraction addition.

Despite the strengths of our method over some prior approaches, the current analysis has several limitations that should be taken into consideration. First, when adaptive problem selection algorithms assign problems to students based on their performance on past problems, the student's performance can itself impact the proportion of violations of sequencing constraints; thus, SCOVA provides correlational, not necessarily causal, information about the impact of orderings. We can avoid this confound by using data with randomized sequences of problems rather than sequences generated from adaptive policies. However, in many cases (as was the case here) we may not have access to randomly generated sequences, and randomized data can often be difficult to collect ethically if we believe that a random sequence could have negative effects on student learning. To test the degree to which this confounds affects our results, we checked if student's pretest scores are correlated with the proportion of violations of various sequencing constraints, which would indicate that students with more prior knowledge tend to adaptively be assigned problems that either obey or violate certain sequencing constraints more than students with less prior knowledge. While we did find such correlations for certain sequencing constraints, the coefficients of the pretest score variables used to predict sequencing constraint violations were less than 0.05 in magnitude, and seemed to indicate that higher-performing students tended to receive ADD earlier and EQ later than lower performing students, which is contrary to the sequences we found most predictive of within-tutor performance! Thus we do not think this confound had a worrisome impact on our results.

Second, ideally we would like to see how sequencing constraints impact student learning as measured via posttest scores rather than just within-tutor performance. However, we were unable to find strong correlations between the proportion of violations of sequencing constraints and the posttest scores of students. This is likely due to the fact that the posttest was comprised of only 16 items and as a result is only a noisy measure of a student's knowledge and does not capture the diversity of concepts taught on the tutor. Note that this is not however a limitation of SCOVA; in theory, SCOVA could be used to compare how various sequencing constraints impact posttest performance.

6. CONCLUSION

We have shown how SCOVA can be used to test a much broader range of sequencing constraints than existing methods (e.g., [13, 21, 19])—including exposure-based, performance-based, interleaving, and blocking constraints. Furthermore, we have shown that when analyzing all of these results in conjunction with each other, a few trends can emerge that can inform practitioners about how to sequence problems. In the case of our fractions tutor, our re-

sults suggest presenting students with fraction equivalence before making and naming on the number line, and presenting the latter before fraction addition. In addition, our results suggest that we should not present the topics in a fully blocked fashion, but rather present four problems of each topic at a time. As for activity types, our results suggest that sense-making should typically come before fluency-building, in agreement with prior literature [14], but that the optimal ordering of activity types may vary for certain fractions topics.

These results suggest just some of the use cases of the SCOVA framework. SCOVA can easily be used to test a broader variety of sequencing constraints, as well as informing old debates about sequencing. For example, prior literature has suggested benefits of interleaving in some cases and of blocking in others [2]. From such results, one may be led to wonder "what is the optimal form of interleaving, and under which circumstances?" While it may be difficult to immediately address such a question in an experimental study, due to the sheer size of the space of sequencing constraints, we can easily analyze such a question using SCOVA.

SCOVA can be of benefit to researchers and practitioners in several ways. First, it can lead to refining hypotheses and determining which questions to test empirically (e.g., testing whether EQ should actually precede MN). Second, it can lead to improving the design of tutor problems (e.g., making EQ problems that use the number line and hence build off of the problems that cover making and naming fractions). Finally, it can help with the construction of adaptive policies (e.g., by determining the order of topics in a mastery learning policy as suggested by performance-based constraints).

7. ACKNOWLEDGEMENTS

The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grants R305A130215 and R305B150008 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Dept. of Education.

8. REFERENCES

[1] W. Battig. Intratask interference as a source of facilitation in transfer and retention. *Topics in learning and performance*, pages 131–159, 1972.

[2] P. F. Carvalho and R. L. Goldstone. The benefits of interleaved and blocked study: different tasks benefit from different schedules of study. *Psychonomic bulletin & review*, 22(1):281–288, 2015.

[3] R. E. Clark, D. Feldon, J. J. van Merriënboer, K. Yates, and S. Early. Cognitive task analysis. *Handbook of research on educational communications and technology*, 3:577–593, 2008.

[4] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *UMUAI*, 4(4):253–278, 1995.

[5] M. C. Desmarais and X. Pu. A bayesian student model without hidden nodes and its comparison with item response theory. *IJAIED*, 15(4):291–323, 2005.

[6] S. Doroudi, K. Holstein, V. Aleven, and E. Brunskill. Towards understanding how to leverage sense-making, induction and refinement, and fluency to improve robust learning. In *EDM*, pages 376–379, 2015.

[7] J.-C. Falmagne, M. Koppen, M. Villano, J.-P. Doignon, and L. Johannesen. Introduction to knowledge spaces: How to build, test, and search them. *Psychological Review*, 97(2):201, 1990.

[8] S. Kalyuga. Expertise reversal effect and its implications for learner-tailored instruction. *Educational Psychology Review*, 19(4):509–539, 2007.

[9] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798, 2012.

[10] K. Korossy. Modeling knowledge as competence and performance. *Knowledge spaces: Theories, empirical research, and applications*, pages 103–132, 1999.

[11] Y. Long and V. Aleven. Supporting students' self-regulated learning with an open learner model in a linear equation tutor. In *AIED*, pages 219–228. Springer, 2013.

[12] M. J. Nathan, K. R. Koedinger, and M. W. Alibali. Expert blind spot: When content knowledge eclipses pedagogical content knowledge. In *Proc. of Cognitive Science*, pages 644–648, 2001.

[13] Z. A. Pardos and N. T. Heffernan. Determining the significance of item order in randomized problem sets. 2009.

[14] M. A. Rau, V. Aleven, and N. Rummel. Complementary effects of sense-making and fluency-building support for connection making: A matter of sequence? In *AIED*, 2013.

[15] A. Renkl and R. K. Atkinson. Structuring the transition from example study to problem solving in cognitive skill acquisition: A cognitive load perspective. *Educational psychologist*, 38(1):15–22, 2003.

[16] F. E. Ritter, J. Nerb, E. Lehtinen, and T. M. O'Shea, editors. *In order to learn: how the sequence of topics influences learning*. Oxford University Press, 2007.

[17] D. Rohrer and K. Taylor. The shuffling of mathematics problems improves learning. *Instructional Science*, 35(6):481–498, 2007.

[18] R. Scheines, E. Silver, and I. Goldin. Discovering prerequisite relationships among knowledge components. In *Proceedings of the 7th International Conference on Educational Data Mining*, pages 355–356, 2014.

[19] S. Tang, E. McBride, H. Gogel, and Z. A. Pardos. Item ordering effects with qualitative explanations using online adaptive tutoring data. In *Proc. of L@S*, pages 313–316. ACM, 2015.

[20] K. Vanlehn. The behavior of tutoring systems. *IJAIED*, 16(3):227–265, 2006.

[21] A. Vuong, T. Nixon, and B. Towle. A method for finding prerequisites within a curriculum. In *EDM*, pages 211–216, 2011.

Measuring Gameplay Affordances of User-Generated Content in an Educational Game

Drew Hicks
North Carolina State University
911 Oval Drive
Raleigh, NC 27606
aghicks3@ncsu.edu

Zhongxiu Liu
North Carolina State University
911 Oval Drive
Raleigh, NC 27606
zliu24@ncsu.edu

Michael Eagle
Carnegie-Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
meagle@cs.cmu.edu

Tiffany Barnes
North Carolina State University
911 Oval Drive
Raleigh, NC 27606
tmbarnes@ncsu.edu

ABSTRACT

Level creation is a creative game-play exercise that resembles problem-posing, and has shown to be engaging and helpful for players to learn about the game's core mechanic. However, in user-authoring environments, users often create levels without considering the game's objective, or with entirely different objectives in mind, resulting in levels which fail to afford the core gameplay mechanic. This poses a bigger threat to educational games, because the core gameplay is aligned with the learning objectives. Therefore, such levels fail to provide any opportunity for players to practice the skills the game is designed to teach. To address this problem, we designed and compared three versions of level creators in a programming game – Freeform, Programming, and Building-Block. Our results show that a simple-to-use building-block editor can guarantee levels that contain some affordances, but an editor designed to use the same core mechanic as gameplay results in the highest-quality levels.

Keywords

User-created Content, Educational Game, Educational Data Mining, Learning Analytics

1. INTRODUCTION

In previous work with our programming game, BOTS, we demonstrated that user-created levels in our game frequently contain appropriate gameplay affordances, which reward specific, desired patterns of gameplay related to the game's learning objectives. Such levels demonstrate the creator's understanding of those learning objectives, and offer other players opportunity to practice using those concepts. However, alongside these high-quality submissions there also exist various negative patterns of user-generated content, four of which we specifically defined in previous work: Sandbox, Griefer, Power-Gamer, and Trivial levels. In various ways, these are levels which ignore or replace the game's core learning objectives and challenges.

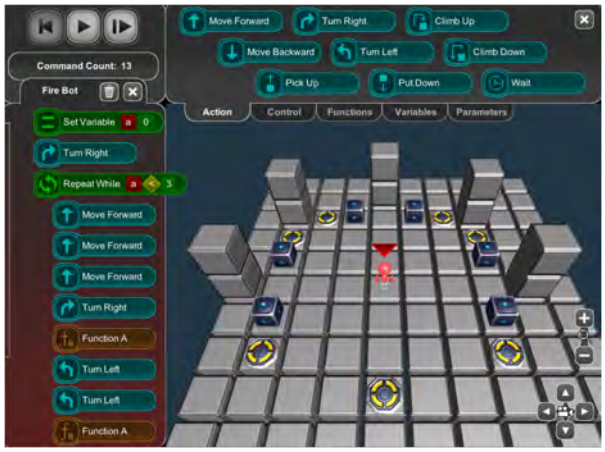


Figure 1: Gameplay screenshot from the BOTS game showing a complex puzzle and partial solution.

In order to implement user-created levels into the game itself, an additional filtering and evaluation step is needed to identify and remove these low-quality submission. Our initial attempt at filtering these levels, a "Solve and Submit" procedure, was effective at reducing the number of these types of levels which were published, and additionally was somewhat effective at reducing the number of these levels created to begin with; however, some users created fewer levels under this condition, indicating that the barrier after level creation discouraged further creation. Our next step is to make further improvements to the content authoring tools in order to increase the overall quality of submitted content.

In order to do so, we will investigate three versions of the game's level editor. The initial, free-form editor, and two constrained editors employing different types of constraints.

Previous work has shown that players are engaged when constraints are posed that are restrictive enough to encourage demonstration of the game's target learning concepts, but not so restrictive as to require them, lest players feel as though they are unable to create what they want to create. We propose to evaluate level editors with two different forms of constraint added. The Programming Editor, where the length (in lines of code) of the solution is constrained, similarly to the Point Value Showcase in Bead Loom Game. Second, where the construction of the level itself is constrained by providing authors with a limited selection of "Building Blocks". For this work, we hope to answer (or gain insight into) the question: Does providing game-like scaffolding, in the form of objectives and points related to elements of high-quality content, result in better user authored content?

2. BACKGROUND

User-generated content has been revolutionizing gaming, and the potential applications in educational games are intriguing. Commercial games such as Super Mario Maker[20] and Little Big Planet[19] rely almost entirely on user-submitted levels to provide an extendible gameplay experience, with the creation process itself serving as the meat of the built-in gameplay. Creative gameplay avoids many of the motivational pitfalls of educational games, such as relying on competitive motivators, that may make the intervention less successful for non-males, who may have a more social orientation towards gameplay, or may have less experience with traditional video games [13, 14, 5].

Creating exercises, in the form of problem-posing, is a common educational activity in many STEM domains. In Mathematics in particular, Problem-posing has been promoted as a classroom activity and as an effective assessment of student knowledge [23, 7]. Games and ITSs such as AnimalWatch[4] and MONSAKUN[17] have users creating exercises for from expert-selected "ingredients." Work with systems such as "MONSAKUN", "AnimalWatch" and the Peer-to-peer learning community "Teach Ourselves" has shown that systems that facilitate problem creation by students can provide benefits beyond those of systems without this feature.

MONSAKUN [17] is a system which facilitates problem-posing for elementary arithmetic problems. The authors wanted to influence students to produce word problems whose structure was different from the structure of the mathematical solution. In order to build the word problem, students are given segments of a word problem such as "Tom has 3 erasers" or "Tom buys several pencils" which they arrange in order to construct their problem.

Animal Watch [1, 4] is a pre-algebra tutor which uses data about exotic animals as the theme for the problems presented. The tutor covers topics such as finding average median and mode, converting to different units, and so on. While the tutor contains around 1000 problems authored by the developers, the authors of this paper noted that even with a large number of problems the system can "run out" of appropriate problems to give a student. The pilot mostly investigated student attitudes towards problem posing, finding that students were excited about sharing content with their peers, and proud that content they had created would be online and accessible to others. At the same time, students reported a low self-assessment of learning, and felt that it was easy once they got started.

Later work by Carole Beal, "Teach Ourselves," investigated these effects further [3], incorporating aspects of gamification. Players earn rewards for solving and creating that are displayed on a leaderboard, and can get "+1" from peers for creating good content in the form of problems and hints. Problems created by students were of usable quality, with an average quality score of 7.5/12 on a scale developed by the system's designers. Teachers who used the system observed increased motivation in their students, and believed that the system encouraged higher-order thinking. Even simple problem-posing interventions have been shown to be effective. In Chang's work with a problem-posing system to teach mathematics, it was demonstrated that when the posed problems were to be used as content for a simple quiz-show-like game, low performing students experienced significantly greater learning gains from the activity, and students reported being more engaged with the activity [8].

3. DESCRIPTION OF BOTS

BOTS (bots.game2learn.com) is a puzzle game designed to teach fundamental ideas of programming and problem-solving to novice computer users. BOTS was inspired by games such as LightBot and RoboRally, as well as the syntax of Scratch and Snap [9, 11, 26]. In BOTS, players take on the role of programmers writing code to navigate a simple robot around a grid-based 3D environment. The goal of each puzzle is to press several switches within the environment, which can be done by placing an object (or the robot itself) on top of them. Within each puzzle, players' scores depend on the number of commands used, with lower scores being preferable. For example, in the first tutorial level, a user could solve the puzzle by using the "Move Forward" instruction 10 times. This is the best score possible without using loops or functions. Therefore, if a player wants to make the robot walk down a long hallway, it will be more efficient to use a loop to repeat a single "Move Forward" instruction, rather than to simply use several "Move Forward" instructions one after the other. These constraints, based on the Deep Gamification framework, are meant to encourage players to optimize their solutions by practicing loops and functions.

Previous work with BOTS focused on how to restrict players from constructing negative design patterns in their levels [16], and how to automatically generate low-level feedback and hints for user-generated levels without human authoring [22, 10]. Our next steps with this game are to further improve the level authoring tools to increase the quality of the levels which don't exhibit these negative design patterns.

3.1 Gameplay Affordances

The term *Affordance* has its origins in psychology, where it is defined by Gibson as "what [something] offers the animal, what it provides and furnishes" [25]. This concept was later introduced to HCI, where Norman defined affordance as "the perceived or actual properties of the thing, primarily those fundamental properties that determine just how the thing

could possibly be used" [21]. Norman's definition centers on users' perspectives. If a user does not read an action with an object possible, then the object does not afford that action.

With respect to affordances in games, James Paul Gee wrote that games create a match between affordances and what he calls "effectivities" [12]. In his writing, *effectivities* are defined as the abilities of the player's tools in the game; for example a character in a platforming game may be able to run, climb, and jump. On the other hand, *affordances* describe relationships between the world and actors, or between tools and actors. Other work taxonomizing level design patterns in video games also referred to the desired gameplay produced by these types of structures. For example, in Hullet and Whitehead's work with design patterns in single-player First-person shooter (FPS) levels, the Sniper Location design pattern is a difficult to reach location with a good view of the play area, occupied by an enemy [18]. This pattern is described as forcing the player to take cover. The presence of other gameplay elements such as Vehicles and Turrets herald similar gameplay changes [2].

In BOTS, the primary educational goal is to teach students basic problem solving and programming concepts such as using functions and loops to handle repetitive patterns. Students (with the robot as their tool) must look at puzzles in terms of opportunities for optimization with loops and functions. Thus, affordances in BOTS come in the form of objects or patterns of objects which both provide and communicate the presence of, these optimization opportunities.

Though the objects in BOTS signal gameplay patterns, players building levels in BOTS frequently place them in misleading or irrelevant ways, where the gameplay decisions informed do not lead to a correct or successful solution. For example, a player can place an extra crate, which communicates that the "Pick Up" command may be used. However, when the optimal solution to the puzzle does not require this crate, the affordance of the crate is meaningless and distracting. Similarly, a player could construct a repetitive structure which affords the use of a "Function" command to navigate, but if ignoring or avoiding the structure entirely results in a better solution, this affordance is also unwanted. Thus, our primary focus is on the subsets of affordances which *involve the core mechanisms in question* relating to problem solving and solution optimization, and through which players can *improve their gameplay outcome* in terms of final score. These are referred to as "Gameplay Affordances" in remaining sections.

3.2 Level Editors

Specific discussion of the design principles behind the two level editors used for this study can be found in our previous work [15]. For the sake of space, we will only generally discuss those design principles here, instead focusing on the tools available to users in the different designs.

In all versions of the level editor, levels consist of a 10x10x10 grid, where each grid square can be populated by a terrain block or an object. Levels must contain at minimum a start point and goal, and can optionally contain additional goals which must be covered with movable boxes before the level will be completed.

Figure 2: The Programming editor interface.

In the Free-Form drag-and-Drop editor, players will be asked to create a level in a Free-Form editor which uses controls analogous to Minecraft. Players can click anywhere in the world to create terrain blocks, and can select objects from a menu such as boxes, start points, and goals, to populate the level with objectives. At any point during creation, the player can save the level (which must, at minimum, contain a start point and a goal.) The player must then complete the level on their own before the level is published and available to other users. In early versions of the Free-form editor, levels began with a 10x10 floor. However, to partially inhibit canvas-filling, this was later changed so that the editor now begins with an entirely blank canvas.

In the Programming Editor (inspired by the Deep Gamification framework [6]) players will be asked to create a level by programming the path the robot will take. To inhibit canvas-filling, players will be constrained to using a limited number of instructions. This is analogous to the level creation tools in BeadLoom Game where players created levels for various "showcases" under similar constraints. This type of constraint has been shown to be effective for encouraging players to perform more complex operations in order to generate larger more interesting levels under the constraints. One challenge with this approach is that since simple solutions are still permitted, and nearly all programs are syntactically correct, users who are experimenting with the level creation interface with no goal in mind may create levels that they themselves do not understand.

In the Building-Block editor, we constrain level creation by providing meaningful chunks to authors in the form of "Building Blocks." This is inspired by problem-posing activities as presented in systems like MONSAKUN [17] and AnimalWatch [1, 4] in which players are asked to build a problem using data and problem pieces provided by experts. In this version of the level editor, players will be asked to create a level only using our "Building Blocks" which are pre-constructed chunks of levels. These "Building Blocks" will be partial or complete examples of the patterns identified in previous work [15], specific structures which correspond to opportunities to use loops, functions, or variables.

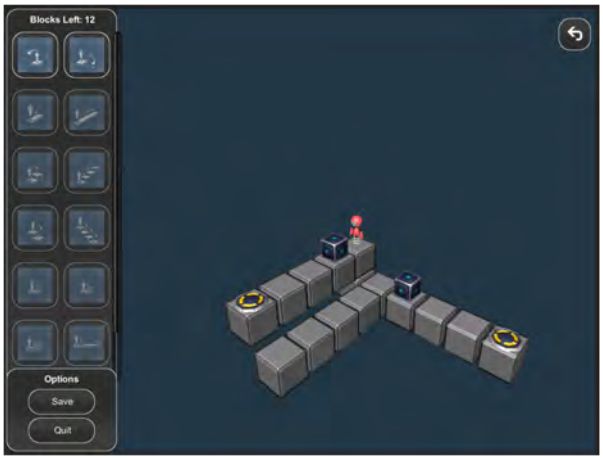


Figure 3: The Building-Block editor interface.

Again, to inhibit canvas-filling, the player is limited to a small number of blocks, regardless of those blocks' size. We hypothesize that this may lead to better levels because it explicitly promotes the inclusion of these patterns, which will lead to opportunities for players to use more complex programming constructs like loops and functions. We also believe that this will encourage students to think about optimizing the solution to the level while they are making the level. One potential challenge with this approach is that students may find these constraints too restrictive, which might reduce engagement for creatively-oriented players [6]. By evaluating these two versions of a gamified level editor against each other, we will determine which practices best suit our game. In particular, which version of the activity leads to the production of better content for future users.

4. DATA

This paper reports gameplay data from 181 unique user IDs (48 in the Programming condition, 61 using Block Editor, 72 using Free-Form Editor) across all classes/workshops that used the BOTS game as part of their activities. In total, 243 levels were created by these players (91 Block / 59 Programming / 93 Free-Form). Of these levels, 9 Block levels and 6 Programming levels were excluded due to bugs in the early versions of the editors rendering them unplayable after their creation, and 3 additional levels (1 Block level, 1 Programming level and 1 Free-Form level) were removed due to other errors, reducing the total number of levels in the sample to 225 levels (81 / 52 / 92). 175 (49 / 33 / 92) of these levels were published and made public. Additionally, after publication the game continually enforces a minimum ideal solution length of 5, automatically setting levels which meet this criteria to be unplayable. After removing these levels, the final count of levels examined by our zero-inflation model was 197 (73, 44, and 80) puzzles, created by 54, 42, and 64 authors. These participants were participants in STEM workshops organized through SPARCS or other outreach activities. Only anonymized game-play data was used for this analysis, to protect participants. For the Free-Form editor, levels from previous experiments were used, as well as anonymous data from other outreach use of the tool, where the same 90 minute session structure was followed.

The additional data was collected in 90 minute sessions, in which all students followed the same procedure. First, each student created a unique account in the online version of the game. Players then completed the Tutorial up to the final challenge level which functions as sort of a "collector" stage; Players aren't expected to complete this level with optimum score, but exploring this level allows faster students to continue practicing while the rest of the class catches up. During the tutorial segment, instructors were told to prompt players to reread the offered hints for their current level carefully, if they became stuck, and only to offer more guidance after the player had carefully read the instructions. This part of gameplay took 45 minutes. Data collected with the Free-form editor used an older version of the game with a longer tutorial. We account for this difference between groups by including tutorial completion in our models.

For the remaining 45 minutes, students were instructed to build at least one level in their version of the level editor interface. After collecting this level, players could continue creating levels, or could play levels created by their peers.

The way the level editor was selected varied per data collection. In the first set of data collections, (data collected prior to the implementation of the new editors) all students used the "Free-Form" level editor to create their levels. To publish their levels, some students were then required to submit a solution to their level before it became public, however this filtering step took place outside of the level editor and after level creation. Therefore, in this data we make no distinction between published or unpublished levels in this condition. One subsequent data collection used only the "Programming" level editor; this data was initially used to evaluate some graphical elements the interface design of that editor. In the remaining data collections, students were randomly assigned an interface between the "Programming" editor and the "Building-Block" editor.

To analyze the differences between created levels, we played each level to find the shortest-path solution from start to goal, and used a solver to find the shortest program to produce this optimal solution. As the actual process of solving a BOTS puzzle would be as complicated as that of a Light-Bot puzzle [24], we used an algorithm which instead, based on student solutions, finds the best optimization of the shortest discovered path in the level. The algorithm used by the optimization solver is a simple: First, a program that recreates the shortest-path using only simple commands is constructed. Then, sets of repeated commands are identified in this program by treating the commands as words and identifying repeated n-grams. Then, recursively, each possible combination of optimization on these n-grams is applied: either replacing the -gram with a subroutine identifier wherever it appears, or replacing adjacent -grams with a single instance of that -gram, wrapped in loop commands. After each step, the program is recursively re-evaluated, until the shortest, most optimal version of the solution is found. The shortest-path solution itself is the *naive solution* which uses only simple commands such as moving and turning. The optimized shortest-path solution is the *expert solution* which uses loops and subroutines to optimize the shortest path solution. The difference between these solutions, in terms of lines of code, is used as a measurement of how well the level

affords the use of those game mechanics.

5. METHODS AND RESULTS

In this section, we describe our analyses, both to identify any differences in the presence of gameplay affordances, and to identify differences in how experts tagged the created levels across conditions.

5.1 Overview of level Improvement

In figure 4 we present the box-plot for score improvement between expert and naive solutions. The light and dark-grey sections are a typical boxplot, showing the median and quartiles of the data. From this, we can see that the zero-value levels are certainly over-distributed (especially in the Free-Form condition) which will impact which statistical methods we use to evaluate this measurement. Additionally, the pink area shows the mean value and the 95% confidence interval around it. From visually inspecting this, we can see that these confidence intervals for the Programming Editor and Free-Form editor do not overlap, implying that the Programming Editor achieves better results. We will confirm this with later analysis.

5.2 Expert Tagging

We compared puzzles across three versions of level editors, with the hypothesis that the more meaningful the level editor's construction unit, the higher quality the puzzles. Here, we assume that "Building Blocks" from version 3 and programs from version 2 are more meaningful than terrain blocks in version 0. We also hypothesize that the Programming editor will result in more reusable puzzles from a player perspective, and that the Building Block editor is more likely to encourage loops and functions.

We used an expert, blind to which editor was used to create the puzzle, tag puzzles, and identify the presence or absence of these negative design patterns. We used the defined puzzle design patterns as identified in our previous work: "Normal" levels which contained few (or no) negative design patterns, and four categories of levels characterized by specific negative design patterns: *Griefer, Power-Gamer, Sandbox, and Trivial* levels, as described in previous work [16].

We measured a puzzle's quality based on previously identified patterns of negative content, which were used as tags for this study. The following criteria were used to assign tags:

- a) it is readily apparent that a solution is possible
- b) a solution actually is possible
- c) the solution can be improved with loops or functions
- d) patterns in the level design call out where loops or functions can be used
- e) the expert solution can be entered in reasonable time
- f) the naive solution can be entered in reasonable time

We decided on these criteria because the pedagogical goal of LOGO-like games, such as BOTS, is to teach students basic problem solving and programming skills. Thus, a good quality puzzle should help players focus on the problem, and should encourage the use of fundamental flow control structures like loops and functions. Levels which are impossible, or simply tedious, are among the most common negative traits identified in previous designs, so updated versions of the level editor specifically addressed these two criteria via hard constraints on the placements of goals and size of levels.

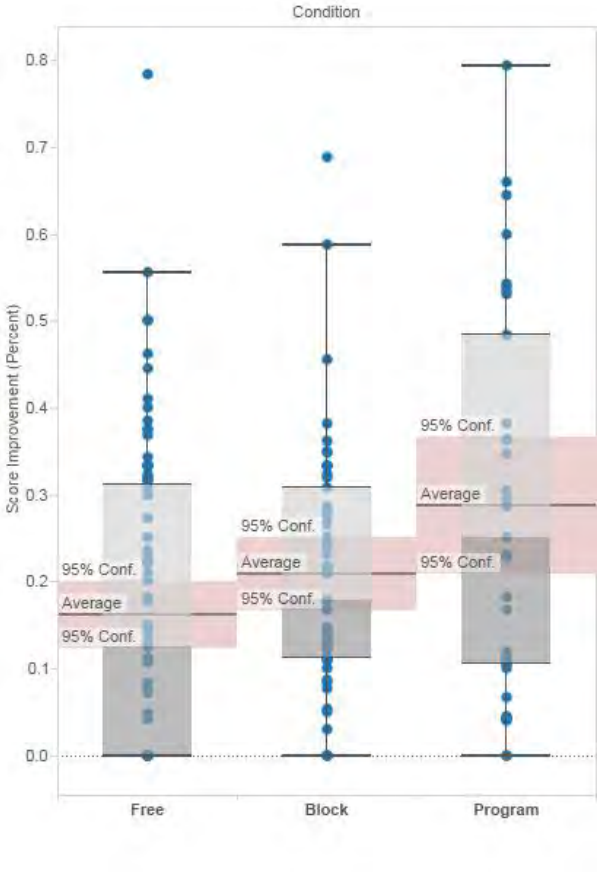


Figure 4: Plot comparing the distribution of levels between the three conditions. Each point in this plot represents the difference in number of commands between a naive and expert solution. In this chart, this is represented as a percentage of the expert solution.

Table 1: Categories of Puzzles Created by Three Versions of Level Editors

	FF	Program	Block
normal	66	43	66
Power-Gamer	9	1	13
griefer	2	0	0
sandbox	10	0	0
trivial	5	8	2
TOTAL	92	52	81

Table 4 reports the number of puzzles in each category, created by the three level editors. Fisher's Exact Test showed a significant difference ($p<.01$) in the category distributions

between each pair of the three level editors.

The Programming editor has the highest proportion of Normal puzzles. Moreover, the Building Block and Free-Form editors created a higher proportion of Power-Gamer levels compared with the Programming editor. These levels are characterized by extreme length and a high number of objectives. The Free-Form editor is the only level editor in which users created Sandbox puzzles, though since our criteria for Sandbox levels include placing off-path objectives and structures (which is quite difficult in the new editors) this is unsurprising. Finally, the Programming editor has the highest proportion of Trivial puzzles.

Since players in the two new editors used a shorter tutorial than players in the free-form condition, we decided to investigate if student performance in this tutorial had an impact on which level editor was more effective. We considered whether or not the authoring player had completed the new tutorial levels during the allotted time. This analysis is again performed on the reduced data set, with levels with solutions less than 5 steps long removed.

5.3 Direct Measurement of Improvement

To further evaluate the differences between levels on a direct measure of possible improvement (the difference in length between a naive solution and an expert solution) we employed a Zero-Inflation model. This type of model is used for modeling variables with excessive zeros and it is usually for overdispersed count outcome variables. Furthermore, it's used when theory suggests that the excess zeros are generated by different process from the other values, and can therefore be modeled independently. Our data indeed has an excess of zeroes, due to the measurement in question, number of lines improved, being a minimum of zero. Additionally, in this case, a level with zero improvement contains no affordances, while a level with only a small improvement may still contain affordances that, though present, are less directly rewarding to the player.

Table 2: Count model coefficients (poisson with log link) comparing the two editors to the baseline, Freeform editor

	Est.	Std. Err.	z value	Pr(>|z|)
(Intercept)	1.905	0.054	35.132	< 0.001 ***
Prog. Editor	0.356	0.076	4.697	< 0.001 ***
Block Editor	-0.031	0.074	-0.413	0.679

Table 3: Zero-inflation model coefficients (binomial with logit link): comparing the two editors to the baseline, Freeform editor

	Est.	Std. Err.	z value	Pr(>|z|)
(Intercept)	-0.568	0.233	-2.436	0.015 *
Prog. Editor	-1.098	0.474	-2.317	0.021 *
Block Editor	-1.067	0.394	-2.705	0.007 **

Presented here are the results of fitting a Zero-Inflated Poisson model on our data. We can look at the two tests separately: the binomial model relates to whether a level will have zero or non-zero results, and the Poisson model relates to the size of the non-zero results. From the binomial model, we can see that the Building-Block editor and Programming editor are more likely than the baseline condition (Freeform Editor) to produce a non-zero result for Difference. This makes sense because, of the Building Blocks available to students, only the very simplest ones offer no affordances, and in fact, the blocks are built out of instances where previous levels contained affordances. So in order to construct a zero-valued level, a Building-Blocks student would need to use only the simplest blocks, though indeed this appears to have been the case in several of the constructed stages. In the Programming editor, the number of commands available are limited, so to make a larger level (as authors tend to do) use of functions or loops is required, and thus the solution to the level will include those same improvements.

Looking at the Poisson model, we see that considering the non-zero results, the Programming editor is likely to have a higher value of Difference than either other condition. In the Building-Blocks editor, each block contains only a small affordance since the blocks themselves are only 3 to 4 commands long. If blocks are not repeated, this pattern will persist in the repeated level. However, in the Programming editor, we observed players exploring more, wrapping code in functions and loops to see what would happen, and changing their code until the level looked how they wanted it to look. Levels generated in this manner will have much larger differences between the naive solutions and expert solutions, than levels generated from multiple unique Building Blocks.

Using a zero-inflated Poisson distribution model, we were able to examine the differences between levels created under our various conditions. We used this zero-inflation model because the model looks for two separate effects: first, the effect that causes the dependent variable to be zero or non-zero, and second, the effect that causes the value of the dependent variable to change in the non-zero cases. This is important because the structural elements for levels with zero affordance for advanced game mechanics are very different from those with only a small affordance—in other words, we would expect the free editor to have more zero-values for the difference between the naive and expert solutions, and the other two editors to have more non-zero values for this difference. Zero-affordance levels tend to be trivially short or entirely devoid of patterns, while small-affordance levels may contain patterns but with small changes between them which limit how advanced game mechanics may be used to optimize the solutions.

To summarize these results, by using this model, we were able to observe the following effects. We first verified our expected result, that both the Programming editor and Building-Block editors are more likely to produce a non-zero result, statistically significantly more likely than the baseline (free-form) condition. The second result is that the Programming editor is likely to have a higher-value difference between and naive and expert solutions, indicating that it promotes puzzles that allow for more optimization.

To investigate if completing the new shorter tutorial had an impact on which level editor was more effective, we considered whether or not the player completed the tutorial levels during the allotted time. The results are presented below:

Table 4: Count model coefficients (poisson with log link) on model, including tutorial completion

	Est.	Std. Err.	z value	Pr(>|z|)
(Intercept)	1.905	0.0542	35.132	< 0.001***
Programming	0.320	0.0813	3.938	< 0.001***
Building-Block	-0.060	0.078	-0.769	0.442
Tut. Complete	0.100	0.078	1.293	0.196

Table 5: Zero-inflation model coefficients (binomial with logit link) including tutorial completion

	Est.	Std. Err.	z value	Pr(>|z|)
(Intercept)	-0.568	0.233	-2.436	0.015 *
Programming	-1.117	0.515	-2.169	0.030 *
Building-Block	-1.083	0.424	-2.556	0.011 *
Tut. Complete	0.054	0.544	0.098	0.922

With this more complex model we see similar results: finishing the shorter tutorial does not have a statistically significant effect, but the coefficient for the magnitude portion of the model is still relatively large. Finishing the tutorial seems to have no compelling impact on the zero portion of the model.

To summarize these results, by using this model, we were able to observe the following effects. First, the Building Block editor is most likely to produce a non-zero result, statistically significantly more likely than either other condition. Second, the Programming editor is likely to have a higher-value of difference for the non-zero results that are created.

6. DISCUSSION

The results seem to confirm that the Freeform editor is the least likely to result in levels with gameplay affordances for using loops and functions. The Freeform editor resulted in the lowest proportion of Normal puzzles, but high proportions of Sandbox puzzles and Power-Gamer puzzles. Additionally, they created fewer puzzles that can be improved by loops or functions, or which have obvious patterns for using loops or functions. Players using this editor are less likely to consider the gameplay affordances of their levels, adding elements regardless of their effect on gameplay. Additionally, the Freeform editor is the only level editor where users created Sandbox puzzles. This may be because Sandbox levels are characterized by the presence of extraneous objects, and the new editors operate by creating the robot's path, so designers would have to deliberately stray from their intended path to place extraneous objects.

On the other hand, the Programming Editor resulted in a high proportion of Normal puzzles and the lowest proportion of Power-Gamer puzzles. This makes sense because a Power-Gamer puzzle is typically a puzzle which takes a short time to create but a long time to complete. Since this editor uses the exact same mechanic for creation as completion, this is quite difficult to do. However, these users also built a lower proportion of puzzles that can be improved with loops and functions than the users of the Building Block editor, and the highest proportion of Trivial puzzles whose solutions are too short to afford the use of loops or functions. This editor is the most complex to use, so players with little patience for learning the interface may create Trivial puzzles. Additionally, trying options at random to see what they do in the programming editor is likely to result in the creation of a Trivial level. We hypothesize that in the other editors, random behavior results in different level types: Power-Gamer levels in the Building Block editor, and Trivial levels in the Programming editor.

Lastly, the Building-Block Editor has a high proportion of normal puzzles, and is slightly more likely to generate a non-zero result than the Programming editor. The building blocks used to create levels are subsections of previously created levels selected specifically because they afford the use of loops or functions. The Building-block editor created the highest proportion of Power-Gamer puzzles. This may be because of the ease of use; adding a block takes one click but may require 5–10 commands from the player who later solves the puzzle. We previously observed that players tended to fill the space available to them in the Freeform editor, so Building-block puzzle creators may also be trying to fill the available space. In both other editors, it takes longer to solve the puzzle than to create it, but the programming editor minimizes this difference, thereby making the creation of Power-gamer levels less likely.

7. CONCLUSIONS AND FUTURE WORK

In conclusion, including Deep Gamification elements in Level Editors (in the form of creative constraints, building blocks, or integration with gameplay mechanic) did result in an overall improvement in level quality. In both the Programming editor and Building-Block editor were more effective than a Freeform editor at encouraging the creation of levels which contain gameplay affordances. The Programming editor was most effective at ensuring a non-zero improvement between expert and naive solutions, but perhaps trivially so, as the building blocks themselves were selected as to contain small improvements. The Programming editor is less likely to ensure a non-zero improvements, but levels created under this condition contain larger improvements, which may be more obvious or more rewarding to players than numerous small improvements.

Our next steps are to investigate how players react to levels created under these conditions. We know that these levels contain opportunities for users to practice, but if the users don't recognize or simply don't take advantage of the opportunities, the improvement is lost. Additionally, we noticed several patterns of negative design that are unique to these new editors, with regards to canvas-filling behaviors. This results in shifting "Sandbox" design into Power-Gamer or Trivial levels. For the new editors, this seems to be mostly negative, resulting in overlong, unrewarding levels. However in the Programming editor, this behavior sometimes resulted in interesting levels created when the author was experimenting with loops and nested functions rather than creating with an end-goal in mind. Similar experimental usage of the previous level editor was treated as negative, with the output levels being low-quality. In the Programming editor, that is not always the case, so re-evaluation of how these levels are identified is needed.

8. ACKNOWLEDGMENTS

Thanks to Michael Kingdon, Aaron Quidley, Veronica Catete, Rui Zhi, Yihuan Dong, and all developers who have worked on this project or helped with our outreach activities so far. This project is partially funded under NSF Grant Nos. 0900860 and 1252376.

9. REFERENCES

[1] I. Arroyo and B. P. Woolf. Students in awe: changing their role from consumers to producers of its content. In *Proceedings of the 11th International Conference on Artificial Intelligence and Education*. Citeseer, 2003.

[2] D. Bacher. Design patterns in level design: common practices in simulated environment construction. 2008.

[3] C. R. Beal, P. R. Cohen, et al. Teach ourselves: Technology to support problem posing in the stem classroom. *Creative Education*, 3(04):513, 2012.

[4] M. Birch and C. R. Beal. Problem posing in animalwatch: An interactive system for student-authored content. In *FLAIRS Conference*, pages 397–402, 2008.

[5] J. Bourgonjon, M. Valcke, R. Soetaert, and T. Schellens. StudentsâĂŹ perceptions about the use of video games in the classroom. *Computers & Education*, 54(4):1145–1156, 2010.

[6] A. K. Boyce. Deep gamification: Combining game-based and play-based methods. 2014.

[7] J. Cai, J. C. Moyer, N. Wang, S. Hwang, B. Nie, and T. Garber. Mathematical problem posing as a measure of curricular effect on students' learning. *Educational Studies in Mathematics*, 83(1):57–69, 2013.

[8] K.-E. Chang, L.-J. Wu, S.-E. Weng, and Y.-T. Sung. Embedding game-based problem-solving phase into problem-posing system for mathematics learning. *Computers & Education*, 58(2):775–786, 2012.

[9] I. F. de Kereki. Scratch: Applications in computer science 1. In *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual*, pages T3B–7. IEEE, 2008.

[10] M. Eagle, D. Hicks, P. III, and T. Barnes. Exploring networks of problem-solving interactions. *Proceedings of the Fifth International Conference on Learning Analytics and Knowledge (LAK 15)*, 2015.

[11] R. Garfield. Roborally. [Board Game], 1994.

[12] J. P. Gee. Deep learning properties of good digital games: How far can they go. *Serious games: Mechanisms and effects*, pages 67–82, 2009.

[13] B. S. Greenberg, J. Sherry, K. Lachlan, K. Lucas, and A. Holmstrom. Orientations to video games among gender and age groups. *Simulation & Gaming*, 41(2):238–259, 2010.

[14] T. Hartmann and C. Klimmt. Gender and computer games: Exploring femalesâĂŹ dislikes. *Journal of Computer-Mediated Communication*, 11(4):910–931, 2006.

[15] A. Hicks, Y. Dong, R. Zhi, and T. Barnes. Applying deep gamification principles to improve quality of user-designed levels. *11th annual Games+ Learning+ Society conference in Madison, WI*, 2015.

[16] A. Hicks, B. Peddycord III, and T. Barnes. Building games to learn from their players: Generating hints in a serious game. In *Intelligent Tutoring Systems*, pages 312–317. Springer, 2014.

[17] T. Hirashima and M. Kurayama. Learning by problem-posing for reverse-thinking problems. In *Artificial Intelligence in Education*, pages 123–130. Springer, 2011.

[18] K. Hullett and J. Whitehead. Design patterns in fps levels. In *proceedings of the Fifth International Conference on the Foundations of Digital Games*, pages 78–85. ACM, 2010.

[19] Media Molecule. LittleBigPlanet. [Video Game], 2008.

[20] Nintendo. Super Mario Maker. [Video Game], 2008.

[21] D. A. Norman. *The psychology of everyday things*. Basic books, 1988.

[22] B. Peddycord III, A. Hicks, and T. Barnes. Generating hints for programming problems using intermediate output.

[23] E. A. Silver. Problem-posing research in mathematics education: Looking back, looking around, and looking ahead. *Educational Studies in Mathematics*, 83(1):157–162, 2013.

[24] A. M. Smith, E. Butler, and Z. Popovic. Quantifying over play: Constraining undesirable solutions in puzzle design.

[25] D. Vyas, C. M. Chisalita, and A. Dix. Dynamics of affordances and implications for design. 2008.

[26] D. Yaroslavski. LightBot. [Video Game], 2008.

The Eyes Have It: Gaze-based Detection of Mind Wandering during Learning with an Intelligent Tutoring System

Stephen Hutt, Caitlin Mills, Shelby White, Patrick J. Donnelly, & Sidney K. D'Mello
University of Notre Dame
384 Fitzpatrick Hall, Notre Dame, IN, 46556, USA
{shutt, cmills4, swhite16, pdonnel4, sdmello}@nd.edu

ABSTRACT

Mind wandering (MW) is a ubiquitous phenomenon characterized by an unintentional shift in attention from task-related to task-unrelated thoughts. MW is frequent during learning and negatively correlates with learning outcomes. Therefore, the next generation of intelligent learning technologies should benefit from mechanisms that detect and combat MW. As an initial step in this direction, we used eye-gaze and contextual information (e.g., time into session) to build an automated MW detector as students interact with GuruTutor – an intelligent tutoring system (ITS) for biology. Students self-reported MW by responding to pseudorandom thought-probes during the tutoring session while a consumer-grade eye tracker monitored their eye movements. We used supervised machine learning techniques to discriminate between positive and negative responses to the probes in a student-independent fashion. Our best results for detecting MW (F_1 of 0.49) were obtained with an evolutionary approach to develop topologies for neural network classifiers. These outperformed standard classifiers (F_1 of 0.43 with a Bayes net) and a chance baseline (F_1 of 0.19). We discuss our results in the context of integrating MW detection into an attention-aware version of GuruTutor.

Keywords

eye-gaze, intelligent tutoring systems, mind wandering, attention-aware learning

1. INTRODUCTION

It is safe to say that most of us have had the experience of reading a text or listening to a lecture and then suddenly realizing that our thoughts have drifted to completely unrelated things, such as an upcoming vacation. This phenomenon, known as mind wandering (MW), refers to the unintentional shift of attention away from the current task towards internal task-unrelated thoughts [32]. MW is a ubiquitous phenomenon, estimated to occur as much as 50% of the time depending on the individual, task, and environment [16].

Not only does MW occur frequently, it can have detrimental influences on performance, especially during educational activities. Indeed, a recent meta-analysis revealed a negative correlation between MW and performance across a variety of tasks, such as lower recall in memory tasks and poor comprehension in reading tasks [24]. It is prudent to point out that MW is not always harmful and the tendency to day-dream has been shown to aid in certain types of tasks, such as creative problem solving [20]. However, research consistently shows that MW impairs performance in tasks requiring concentrated attentional focus and integration of information from the external environment as is the case with many learning activities [21].

Considering the negative influence of mind wandering on learning [27, 29, 30], it is important to take steps towards developing intelligent systems that help reorient attention to assuage the negative effects of MW. This requires an ability to monitor the locus of attention, detect students' current attentional state, and provide a stimulus to direct focus back to the learning task [10]. Detecting MW is no easy task however. Although MW is related to other forms of disengagement, such as boredom, behavioral disengagement, and off-task behaviors [1, 2, 9, 18, 36], it is inherently distinct because it involves internal thoughts rather than overt expressive behaviors. This raises two challenges. First, while other disengaged behaviors often involve detectable behavioral markers (e.g., yawns signaling boredom), mind wandering is an internal state that can look similar to on-task states. Secondly, the onset and duration of MW cannot be precisely measured because MW can occur outside of conscious awareness.

Despite these challenges, there has been some progress toward automatic detection of mind wandering during reading (discussed as related works in Section 1.1). However, almost all of the current MW detectors focus on reading tasks, so their effectiveness is unclear during complex interactive tasks, such as learning with advanced learning technologies. Here, we explore for the first time, automated approaches for MW detection during learning with intelligent tutoring systems (ITS).

1.1 Related Work

In an early study attempting to detect MW in the context of learning [11], students were asked to read a paragraph about biology aloud, followed by either self-explanation or paraphrasing. Students self-reported how frequently they zoned out on a scale from 1 (all the time) to 7 (not at all). A supervised machine learning model trained on acoustic-prosodic features to classify low (1-3 on the scale) and high (5-7 on the scale) zone outs achieved an accuracy of 64%. However, it is unclear whether this detector could generalize to new students as the validation method did not ensure student-level independence across training and testing sets.

Some researchers have built MW detectors based on information readily available in log files collected during the reading (e.g., reading time, complexity of the text). For example, [19], attempted to classify whether students were MW while reading a screen of text using reading behaviors and features of the text,

such as text difficulty. They were able to classify MW at 21% greater than chance using a leave-one-subject out cross-validation method. Similarly, another study [12] also attempted to predict MW during reading using textual features, such as word familiarity, difficulty, and reading time. However, rather than using supervised machine learning, they used a set of researcher-defined thresholds to ascertain if participants were "mindlessly reading" based on difficulty and reading time.

More recent studies have explored additional techniques to detect MW during self-paced computerized reading [5, 7, 12, 19]. In these studies, MW was measured via thought probes that occurred on pseudo-random screens (i.e. screen of text similar to a page of text). Participants responded either "yes" or "no" based on whether they were MW at the time of the probe. Supervised classification models were trained to discriminate the two responses using physiological features (e.g., skin conductance, temperature) [7] or eye-gaze [9], achieving accuracies ranging from 18% to 23% above chance and validated in a manner that generalized to new students. Further, combining the two modalities led to a 11% improvement in detection accuracy above the best individual modality [3].

Previous attempts to detect MW from eye-gaze are of particular relevance to the current paper. Eye tracking offers a unique possibility to automatically detect MW due to well-known relationships between visual attention and eye-movements. For example, MW has been associated with longer fixation durations [26] and more blinking in reading [33]. These and other relationships have been leveraged to build MW detectors during reading [4, 6] with moderate levels of success. However, it is unclear if these findings and corresponding detectors generalize to other activities, particularly activities where eye-gaze does not have the predictable patterns found in reading text.

1.2 Current Study and Novelty

The primary focus of this paper is to detect MW during learning with an ITS called GuruTutor. Previous work suggests that MW occurs, on average, once every two minutes during interactions with GuruTutor and is negatively correlated with learning gains [17], highlighting the importance of detecting MW in this context.

There are a number of novel aspects with this work. First, we study MW detection in an interactive context– an ITS with conversational dialogues and other embedded activities. Detection of MW during interactions with an ITS provides additional challenges compared to reading. In reading tasks, it is generally clear where the reader should be looking if they are engaged in the task and the eyes move across the screen in a predictable manner. However, in complex environments such as an ITS, there are far more paths the eyes may take, resulting in fewer predictable patterns, rendering MW detection more difficult.

Second, GuruTutor includes multiple activities, such as lecturing, scaffolded dialogue, concept mapping, and Cloze task completion. Each has a different visual layout, level of interactivity, and learning goal, presumably engendering different gaze patterns and levels of MW. By requiring our MW detector to work across a range of activities, we hope to have a solution that will generalize to additional learning technologies that may support quite different activity types.

Third, while researchers have typically used standard classification algorithms (e.g., Naïve Bayes, decision trees), we explore the use of a genetic algorithm (GA) to evolve neural networks (both topologies and connection weights) for detecting MW. This approach evolves the weights and topology concurrently, thereby implicitly integrating feature selection and feature weighting. Further, MW detection suffers from a data-imbalance problem in that the standard classifiers are skewed towards predicting the majority class, which is typically the class associated with Not MW. We address this issue by considering various GA fitness functions that focus on balancing the precision and recall of the minority MW class.

Fourth, we use a low-cost consumer-grade eye tracker to collect gaze data from participants as they interact with Guru. Research grade eye trackers can cost upwards of $40,000, so the use of affordable equipment (less than $150) increases the scalability of the detector for eventual deployment in real world learning environments such as computer-enabled classrooms.

2. DATA COLLECTION

We adopted a supervised classification approach for MW detection, which entailed collection of training and validation data.

2.1 Participants

Participants were 105 undergraduate students (69.5% female, average age 19.14) from a mid-sized, private university in the Midwest. Participants received extra credit or course credit for participating in the study.

2.2 GuruTutor

GuruTutor (Guru) is an ITS designed to teach biology topics through collaborative conversations in natural language. It is modeled after interactions with expert human tutors [22]. Guru engages the student through natural language conversations with an animated tutor agent that references a multimedia workspace, animating content relevant to the conversation (see Figure 1). Students type in responses in a conversational style that Guru analyzes using natural language processing. Guru maintains a student model which it uses to tailor instruction to individual students. Guru has been shown to be effective at promoting learning and retention at levels similar to human tutors [22].

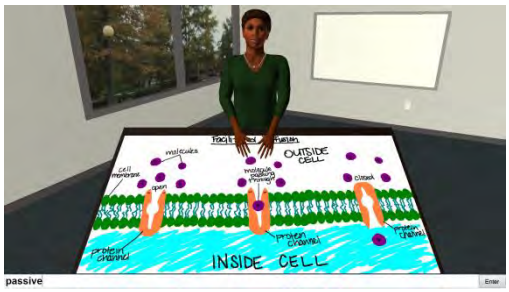


Figure 1. Example of Guru during CGB Phase

Guru presents biology topics aligned with state curriculum standards (e.g., *cellular respiration*), typically lasting between 15 to 40 minutes each. Each topic contains sets of interrelated concepts and facts (e.g., *proteins help cells regulate functions*). Guru begins each new topic with a brief preview to introduce it to the student, followed by a five phase session that encourages students to build and articulate their understanding of the concepts. These five phases are described below.

Common-Ground-Building Instruction (CGB Instruction). Biology lessons often involve specialized terminology that needs to be well understood before it is possible to move on to more collaborative knowledge building activities. Therefore, Guru

begins with a collaborative lecture phase that covers basic information and terminology relevant to the topic. **Intermittent Summaries (Summary).** Following CGB, students generate summaries using natural-language to describe the content covered. These summaries are automatically analyzed to determine which concepts to target throughout the remainder of the session. **Concept Maps**. For the target concepts, students complete skeleton concept maps, node-link structures that are automatically generated from concept text. **Scaffolded Dialogue.** Next students complete a scaffolded natural language dialogue in which GuruTutor uses a Prompt → Feedback → Verification Question → Feedback → Elaboration cycle to cover target concepts. If a student shows difficulty mastering particular concepts, a second Concept Maps phase is initiated followed by an additional Scaffolded Dialogue phase. **Cloze Task.** The session concludes with a cloze task requiring students to complete an ideal summary of the topic by filling in blanks to connect key words to related concepts.

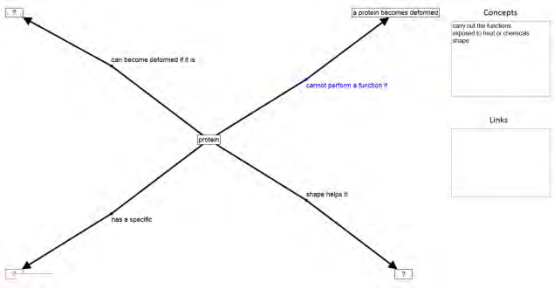


Figure 2. Example of Guru during Concept Maps

2.3 Procedure

All experimental procedures were reviewed and approved by the university's ethics board. After signing an informed consent, participants were seated at a desk in front of a 15-inch laptop. A Tobii EyeX eye-tracker was positioned directly under the laptop screen using a magnetic strip based on the guidelines provided by Tobii.

Participants were asked to sit comfortably with the chair pulled up to the desk. Next, participants were given an explanation of MW and were given detailed instructions for how to respond to the mind wandering probes (see below) during learning with Guru. Specifically, MW was defined as "when you realize that you are no longer paying attention to what you're supposed to be doing, for example, instead of thinking about the biology, you may be thinking about something else altogether."

After receiving initial instructions, a 60 second calibration process occurred before beginning the learning session. Participants were dynamically instructed about their seating and head position in order for the eye tracker to pick up their eye gaze.

Then, one of six biology topics from Guru was assigned to each participant: Interphase, Osmosis, Biochemical Catalysts, Carbohydrate Function, Protein Function, or Facilitated Diffusion. Following a pretest on the assigned topic, participants began the Guru tutoring session. Afterwards, participants completed a posttest and were fully debriefed.

2.4 Mind Wandering Probes

Mind wandering was measured during learning with Guru using auditory thought probes, which is a standard approach in the literature [31]. Participants were probed at pseudo-random intervals with probes occurring every 90-120 seconds, this was based on previous work investigating how often MW occurs[17]. If the tutor was speaking at the time the probe was triggered, the probe was paused until the tutor finished speaking so as to not interrupt the conversation flow. Probes consisted of an auditory beep that automatically paused the tutoring session. An opaque overlay would then appear on screen, instructing the participant to press the "N" key if they were not mind wandering, the "I" key if they were intentionally (deliberately) mind wandering, or the "U" key if they were unintentionally (spontaneously) mind wandering. In this study, we do not differentiate between intentional and unintentional mind wandering, and "I" and "U" responses were coded as "MW" to indicate mind wandering occurred. Participants encountered an average of ten probes over the course of the session. We obtained a total of 1104 reports to thought probes, 17% of which corresponded to episodes of MW.

3. MODEL BUILDING

Supervised machine learning models were built to detect MW using eye-gaze data and contextual information from Guru.

3.1 Feature Engineering

We calculated features from a short window of time preceding each auditory probe, exploring window sizes ranging from 3 to 30 seconds. We did not consider windows shorter than 3 seconds, as they most often did not contain sufficient gaze data. We discarded windows where not all the eye-gaze features could be computed, such as cases when the face was occluded or the student was looking down at the keyboard. For the smallest window (three seconds) 418 instances were removed, lowering the MW rate to 15.5%. A total of 156 instances were removed for all other window sizes, leaving the average MW rate unaffected (17%).

Table 1. Eye-gaze features

Fixation Duration	duration in milliseconds of fixation
Saccade Duration	duration in milliseconds of saccade
Saccade Length	distance of saccade
Saccade Angle Absolute	angle in degrees between the x-axis and the saccade
Saccade Angle Relative	angle of the saccade relative to previous gaze data.
Saccade Velocity	Saccade Length / Saccade Duration
Fixation Dispersion	root mean square of the distances from each fixation to the average fixation position in the window
Horizontal Saccade Proportion	proportion of saccades with angles no more than 30 degrees above or below the horizontal axis
Fixation Saccade Ratio	ratio of Fixation Duration to Saccade Duration

Note. Bolded cell indicates that the total number, mean, median, min, max, standard deviation, range, kurtosis, and skew of the distribution of each measurement were used as features.

Gaze Features. Eye movements are measured by fixations (i.e. points in which the gaze was maintained on the same location) and saccades (i.e. the movement of the eyes between fixations). We calculated fixations and saccades from the raw eye-gaze data using the Open Gaze and Mouse Analyzer (OGAMA) [35], an open source package for eye tracking analysis. Next, gaze features were computed for each from the fixations and saccades (see Table 1) in that window. We considered six general measures based on fixations and saccades. For these gaze measures, we calculated the number, mean, median, min, max,

standard deviation, range, kurtosis, and skew of the distributions of each measure across the time window, yielding 54 features. We also included three other features (listed in Table 1), yielding a total of 57 gaze features.

Contextual Features. The gaze features were complemented with eight contextual features that provide a snapshot of the student-tutor interaction context during each window. One feature was the assigned biology *topic*. A second encoded participants' *pretest* scores on that topic. The next three of these features describe participants' progress within Guru, such as the *current phase* of the session (e.g., cloze, concept map, etc.), the amount of elapsed *time into the session*, and the amount of elapsed *time into the current phase*. The last three context features focused on participants' overall interaction with Guru, measured by the amount of positive, neutral, and negative feedback received.

3.2 Addressing Class Label Imbalance

Only 17% of the 1104 thought probes were reports of MW, thereby leading to substantial data skew. This imbalance between the class labels poses a challenge as some supervised learning methods tend to bias predications towards the majority class label. To compensate for this concern, synthetic oversampling was applied to provide a more balanced class distribution on the training set only. The SMOTE algorithm [8] creates synthetic instances of the minority class by interpolating feature values between an instance and randomly chosen nearest neighbors. No SMOTING was done on the testing set in order to ensure validity of the predictions.

3.3 Classification Models

We evaluated five classifiers frequently explored for the detection of MW [6, 7]. These included Bayesian networks, logistic regression classifiers, multilayer perceptrons (MLP), random forests, and support vector machines (SVM) using implementations from the WEKA data mining software [14].

We also considered a neural network trained using a genetic algorithm (GA), which is a type of evolutionary algorithm for optimization and search problems that uses techniques loosely inspired by biological natural selection. GAs maintain a population of candidate solutions (phenotypes), each with a set of properties (genotypes). These individual solutions evolve over time guided by a fitness function. At each generation, the fitness function is used to rank the candidate solutions, allowing elimination of inferior solutions and selection of the best candidates to the new generation. New candidate solutions are created at each generation through the mechanisms of mutation, a pseudo-random perturbation of an individual's genotype, and cross-over, the combination of aspects of the genotypes of multiple fit individuals.

NEAT Algorithm. In this study, we used a GA to evolve an artificial neural network for MW detection. We used the NeuroEvolution of Augmenting Topologies (NEAT) algorithm to evolve the topology of neural network alongside an evolution of the network weights [34]. Because NEAT evolves both the weights and topology of the network, it must implement the genetic operators of mutation and crossover in a unique way to handle differences between network topologies. NEAT uses population speciation to track individuals with similar topologies, restricting crossover to individuals with similar network topologies to ensure the resulting new topology is coherent. Mutation of the topology occurs in two ways, either by the creation of a hidden node or the addition or removal of a link between nodes. As the size of the networks may grow larger in each new generation, constraints are imposed to penalize large networks that exceed a complexity threshold.

To encourage innovation in new generations, NEAT implements speciation by grouping networks that share similar topologies into the same population. The populations are determined by a distance metric that computes the distance of a topology of an individual from the initial topology of the species. New populations are created as new networks that are dissimilar from any existing population evolve. This strategy allows the generation of new individuals by applying genetic operators on similar individuals in order to maintain viable network topologies without hindering the ability of the GA to develop new and unique networks.

Using NEAT for MW Detection. We used SharpNeat, a popular implementation of the NEAT algorithm in the C# language [28]. We tuned the evolution variables on our data in preliminary experiments. We used a population of 150 individuals and ran the algorithm for 500 generations. We also determined a complexity threshold to prune overly complex networks. Because evolutionary algorithms are non-deterministic, we ran these classifiers over multiple iterations in each experiment.

The effectiveness of an evolutionary algorithm depends on the evaluation of individuals using the fitness function. We considered three different fitness functions that were informed by [13]. The first function evaluates candidate networks using the overall accuracy (recognition rate) of the model. The second function evaluates the networks considering the F_1 measure for the class label of interest, which in our case is MW (denoted as F_1-MW). The third evaluates the networks using the Youden's J-statistic, (a variation on Cohen's Kappa, sometimes called "informedness" [23]) which is defined as *sensitivity + specificity – 1* of MW.

3.4 Cross-Validation

All experiments were conducted using leave-several-participants-out cross-validation. For each iteration of the classifier, instances from 66% of the participants were assigned to a training set and the remaining instances of the other 33% participants were assigned to a test set. This process ensures that no instances of any individual participant could appear in both the training and test sets within a fold. This process was repeated for 15 folds, and the results accumulated. We selected 15 iterations in order to balance time taken to build the models (as evolutionary approaches are slow) and reliability by testing multiple training/testing set pairs. Minority oversampling (SMOTING) occurred within each fold and on the training set only.

4. RESULTS

We report the F_1 measure in our evaluation of our results. This measure is common in information retrieval tasks and provides a balance between precision and recall. Because our intention is to detect instance of MW, we focus on the F_1 score of the MW label as our key metric. This is a very strict evaluation criterion as the base rate of MW is only 17% in our data. To facilitate comparisons with previous (and future work), we also reported the F_1 score for the majority Not MW class (83% of instances), as well as the weighted F_1 score.

4.1 Comparing Window Size

In our first experiment, we explored the influence of various window sizes ranging from 3 to 30 seconds. As we are interested

in general trends, we average results of the five standard classifiers and the three NEAT classifiers. (see Figure 3). These results illustrate a general trend of improved performance for the larger windows, although these differences may not be overly large. In the remainder of this work, we considered a 30 second window in our experiments as it generally resulted in the highest F_1 scores.

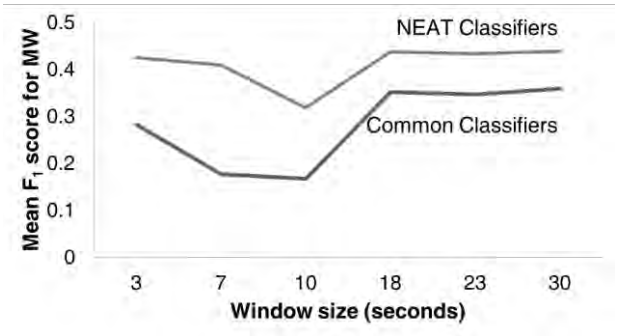


Figure 3. Comparison of different window sizes.

4.2 Comparison of Classifiers

In Table 3 we report the results of the classifiers considering a 30 second analysis window, informed by our experiment in Section 4.1. The highest F_1 for MW is denoted in bold for both the common classifiers and NEAT implementations that varied by fitness function. For comparison, a chance-level baseline was created by randomly assigning a class label to each instance based on the observed MW rate of 17%. We note that all of the classifiers showed an improvement in detecting the target minority class of MW over the chance model.

Table 2. MW detection results by classifier for 30 second window

	F_1 of MW	F_1 of Not MW	Overall F_1
Standard Classifiers			
Bayesian Network	**0.43**	**0.73**	0.68
Logistic Regression	0.38	0.79	0.72
MLP	0.30	0.83	0.74
SVM	0.37	0.76	0.70
Random Forest	0.23	0.86	0.75
NEAT Classifiers			
Fitness: Accuracy	0.36	0.76	0.69
Fitness: F_1-MW	**0.49**	**0.58**	0.57
Fitness: Youden J	0.44	0.69	0.65
Baseline	0.19	0.83	0.73

Among the common classifiers, Bayesian network achieved the highest F_1 score for MW. This was also the case in previous MW eye-gaze detectors in other domains [6]. The overall F_1 score for the Bayesian network was lower than for other classifiers, ostensibly because the other classifiers tend to over predict the majority class. For NEAT, using the F_1-MW score as the fitness function resulted in the overall best F_1 score for MW. NEAT with Youden's J- statistic as the fitness function did yield a slightly more balanced detector with an increase in F_1 of Not MW. Importantly, the best NEAT classifier outperformed the Bayesian network at detecting MW, which is our target class of interest. In Table 3 we show the confusion matrices for the three classifiers that obtained the highest F_1 score for MW: the Bayesian network, NEAT-F_1-MW, and NEAT-Youden. NEAT-F_1-MW yielded a substantially higher hit rate than the other two classifiers, but also suffered from a high false positive (FP) rate. The Bayesian network and NEAT-Youden had similar patterns of errors in that they had both lower hit rates as well as FP rates. Based on these results, we consider NEAT-F_1-MW and the Bayesian network in subsequent analyses.

Table 3. Confusion matrices for the three best classifiers

Actual	**Predicted**	
Bayes Net	MW	Not MW
MW	0.52 (hit)	0.48 (miss)
Not MW	0.34 (false pos.)	0.66 (correct rej.)
NEAT-F_1-MW	MW	Not MW
MW	0.69 (hit)	0.31 (miss)
Not MW	0.54 (false pos.)	0.46 (correct rej.)
NEAT-Youden	MW	Not MW
MW	0.55 (hit)	0.45 (miss)
Not MW	0.41 (false pos.)	0.59 (correct rej.)

4.3 Gaze only vs. Gaze + Context Features

We investigated the utility of contextual features over the gaze features alone (see Table 4). The addition of contextual features improved the F_1 score for the minority class of MW for NEAT and correspondingly for the majority Not MW class for the Bayesian network. Overall, the improvements in performance were small, suggesting that the gaze features were more important to the detection of MW compared to the contextual features.

Table 4. Gaze (G) vs. Gaze + Context (G+C) features

Classifier	**Feature**	**F_1 of MW**	**F_1 of Not MW**	**Overall F_1**
Bayesian network	G	0.45	**0.69**	0.65
	G+C	0.43	**0.73**	0.68
NEAT-F_1-MW	G	**0.44**	0.58	0.56
	G+C	**0.49**	0.58	0.57

4.4 Oversampling vs. No Oversampling

In Section 3.2, we discussed the imbalance between instances of MW and Not MW in the dataset, and addressed this difficulty by supplementing the training data with the SMOTE oversampling technique. To study the effect of SMOTE, we compared the Bayesian network and the best NEAT classifier on datasets with and without these synthetic training instances (see Table 5). We confirmed that synthetic oversampling indeed improved the classification of the MW (the minority class) for NEAT at the cost of detecting the majority class. Thus, SMOTING played a critical role in reducing the tendency to over predict to the majority class. SMOTING had no notable effect for the Bayesian network, which seemed to be more impervious to data skew.

Table 5. Results with and without oversampling.

Classifier	SMOTE	F_1 of MW	F_1 of Not MW	Ove all F_1
Bayesian net	No	0.41	0.75	0.70
	Yes	0.43	0.73	0.68
NEAT-F_1-MW	No	**0.42**	**0.75**	0.79
	Yes	**0.49**	**0.58**	0.57

4.5 Analysis of Features

Neural networks use a mathematical approach to transform and combine input features to useful output. Thus, we can learn more about the structure of our MW detector by investigating the topologies formed during the evolutionary process. For example, a network with a densely connected hidden layer would be performing a large amount of internal calculations compared a sparsely connected layer.

To better understand our MW detector's structure, we examined each of the 15 iterations of the *NEAT-F_1-MW* model and investigated the networks that survived to the final generation in each case. Across the networks the mean number of hidden nodes in the network is 1.6 (min 0, max 3), the average number of inputs actually used in the final network is 17.133 (min 8, max 36) and the average number of connections is 21.46 (min 9, max 44). The number of hidden nodes here is low, but considering the large number of inputs to a small number of outputs, this is to be expected. The algorithm also biases towards smaller networks to avoid bloat.

Developing neural network topologies also provides inherent feature selection that takes place as the network structures evolve to subsequent generations. This provides an opportunity to explore which features were most useful in detecting MW. Seven features appeared in at least half of the final networks as shown in Table 6.

Table 6. Cohen's *d* of most commonly used features

Feature	Cohen's *d*
Fixation Duration Skew	-0.27
Minimum Fixation Duration	0.17
Mean Saccade Duration	0.32
Saccade Duration Kurtosis	-0.16
Saccade Duration Skew	-0.17
Minimum Saccade Velocity	-0.15
Fixation to Saccade Ratio	-0.17
Pre Test Score	-0.18

We compared these seven features across the MW and not MW instances using an effect size measure (Cohen's *d*). An effect size measure is appropriate for this comparison in order to evaluate the direction and magnitude of the differences between the two classes. Positive values depict higher values for instances of MW (see Table 6). In general, the differences reported in this paper are consistent with previous work examining eye gaze surrounding MW episodes during reading [4]. Two of the seven features had differences across the MW and not MW classes consistent with small effect sizes ($|d| > .2$). The largest difference was seen for mean saccade duration ($d = .32$). This finding suggests that participants tend to have longer gaps between fixations leading up to a MW episode as opposed to more rapid eye movements between fixations. A similar effect size was found for fixation duration skew ($d = -.27$), which suggests that there is a higher probability that participants would have shorter fixations before a MW episode occurs compared to when their attention is on task.

It is important to point out that the low Cohen's *d* values ($< |.2|$) are not entirely surprising given the nature of neural networks. The network employs a combination of features and the combination sets that prove to be most effective for MW detection may not be consistent with the overall largest mean differences. Instead, the important thing to note is that these seven features were the most consistent across all iterations.

It is also worth mentioning that only one context feature was present in over half of the final networks: pre-test score. Instances of MW were associated with lower pre-test scores, indicating that when participants were more likely to mind wander if they did not understand the topic well to begin with.

5. GENERAL DISCUSSION

Mind wandering occurs frequently during learning and has a negative impact on learning outcomes [21]. An attention-aware learning technology [10] that can automatically detect MW could intervene to re-engage learners, assuaging the cost of MW on comprehension to improve learning. However, MW is a covert, internal state with no obvious behavioral markers, making it difficult to detect. Although strides have been made to detect MW in the context of self-paced reading, MW detection has not yet been attempted in the context of an ITS – a challenge we addressed in the current paper. In the remainder of this section, we discuss our main findings, consider potential applications, and discuss limitations and future work.

5.1 Main Findings

MW detection during reading tasks is supported by decades of research on MW and eye movements [25]. However, more complex learning interfaces, such as the ITS used here, are not afforded such predictable patterns of eye movements. Despite these challenges, we demonstrated the ability of a neural network trained using a GA to detect MW in the context of learning with an ITS. We were able to accurately classify MW with an F_1 of 0.49 at detecting the minority MW class. Although this result is modest, it is an important first step in detecting MW in this novel domain.

In most machine learning tasks, a large imbalance in the distribution of class labels results in a degraded performance at predicting the minority class label [15]. This is a major issue for MW detection as its rate of occurrence is around 20% to 40% in learning contexts [27] and in our case it was 17%. We addressed the data imbalance by using a synthetic oversampling technique and by tweaking the fitness function of the GA in order to help the classifiers in detecting the minority class of MW. We believe that this combined approach might be beneficial for other classification problems when there is severe data skew.

Since MW detection in the context of learning from an ITS is still in its infancy, it was important for us to adopt a method that will generalizable for future work in this area. The eye gaze feature set was limited to eye movements that were independent of the specific content being displayed on the screen. This enabled our models to operate across Guru's multiple instructional activities, each with very different visual displays.

In addition to the gaze features, a second set of features included the context of the learning session. A comparison of model performance with and without contextual features revealed that contextual features added a small, but not substantial, improvement in detection accuracy. This finding further illustrates the idea that eye gaze can be a powerful signal of attention, regardless of the learning context.

An analysis of the most consistent features in the model point to seven important features, six of which are gaze features. MW episodes had a longer mean saccade duration, yet smaller fixation duration skew. The longer mean saccade duration preceding MW is consistent with prior research, which suggests that MW signals a breakdown at very basic levels of perceptual processing [30] – in this case, being slower to direct your eyes form one point to another. Most of the effect sizes (*d*'s) reported are objectively small effects; however, we feel that obtaining a sense of consistent features and how the relate to MW is a major contribution at this stage in the of MW detection.

All data was collected using low-cost, consumer-grade eye trackers (less than $150). This is a marked contrast compared to many research-grade trackers that can cost tens of thousands of dollars. Our goal is eventual deployment of our models at scale, thereby allowing us to test generalizability in more diverse contexts. For this reason, it was important to ensure that our models were validated in a student-independent manner, which increases our models' ability to generalize to new students. Taken together, these results increase our confidence that the models will generalize more broadly, though this claim requires further empirical validation.

5.2 Applications

The key application of this work is to develop an attention-aware version of Guru that detects and combats MW in real-time. Once the goal of MW detection is realized, Guru has a number of paths to pursue to re-engage attention.

At an immediate level, one initial effect of MW is that the student simply fails to attend to a unit of information or a salient event in the learning environment. The unattended information, question, or event is needed to construct an adequate mental model so that subsequent knowledge can be assimilated or the student will be left behind. Thus, a simple direct approach is to reassert the missed information ("e.g., Mary, let me repeat that….") or highlight the information by directing attention to specific areas of the display (e.g., "Mary, you might want to look at the highlighted image showing the chromosomes duplicating"). Taking a somewhat different approach, Guru can also launch a sub-dialogue where it asks a content-specific question (e.g., "Mary, what happens to the chromosomes when they duplicate") or asks the student to complete a mini-activity (e.g., "Mary, we now have a simulation of the first phase in mitosis. Can you…."). Guru can also ask the student to self-explain when MW is detected.

Additional measures might be needed if MW persists despite these intervention strategies. One option is to simply change to a new activity. Guru might even suggest changing topics or offering a choice for what students would like to do next. If all else fails, Guru might even suggest that the student take a break.

It is important to note that the proposed intervention strategies rely on MW detection, which is inherently imperfect. The detector might inaccurately assert that a student is MW when they are not (false alarms) or it might assert that a student is actively attending when they are in fact MW (misses). MW detection does not need to be perfect as long as we account for this in MW interventions. For example, Guru can adopt a probabilistic approach where the MW detector provides an estimate of the likelihood that the student is MW. This likelihood will guide whether an intervention is launched (i.e. if the likelihood of MW is 70%, there is a 70% chance that an intervention will be triggered). Second, interventions can be designed to be "fail-soft" in that there are no harmful effects if delivered incorrectly.

5.3 Limitations and Future Work

There were several limitations with this study. One key limitation pertains to the moderate MW detection accuracy. Although, we detected MW above chance levels using several different classifiers, these results leave room for improvement. Ongoing work seeks to reduce the false positive rate while increasing the hit rate for our MW models by expanding our feature set and incorporating temporal information in the machine learning.

We designed our approach to include a low-cost eye tracker, however, these consumer models have a lower sampling-rate, limiting the accuracy of the eye-gaze data compared to research-grade eye trackers. Furthermore, although we desire to eventually deploy our system in noisy classroom environments, we only tested our system in a quiet lab setting.

This work is also limited by the features used in the supervised learning process, which were a small and potentially restrictive subset of gaze features. We also did not model temporal patterns of eye movements, such as examining if the participant revisited an area of the screen they had previously viewed. Additionally, we only used a small number of contextual features. Future work may consider a utilizing log files from the tutoring session more extensively to create more in-depth context features (e.g., content, timing, and length of student responses, etc.).

The results of this study invite several avenues for improvement which we will explore as future work. First, we will explore additional eye-gaze features, such as those that track localized regions of interest but at a level of abstraction that does not limit generalizability to additional interfaces. Informed by our observation that the inclusion of contextual features improved detection of MW, we will explore additional contextual features from the ITS, again with an eye for more generalizable features (e.g., response time). Furthermore, it is possible to build multiple MW detector specialized for different phases in the Guru tutoring sessions, although this would require a large amount of data and would make these detectors less able to generalize to other ITSs. Finally, we will collect data in the real-world context of a computer-enabled classroom where 20-30 students interact with Guru on individual computers while their gaze is being tracked. Indeed, preliminary data collection on this front is already underway.

5.4 Concluding Remarks

Attention is a crucial part of learning. An attention-aware ITS that can detect a student's attentional state as well as redirect their attention to better engage them in the learning task could be very beneficial for engagement and learning. Attention-awareness, however, requires monitoring of attention, which has historically been limited to the lab. However, advances in consumer-grade eye-tracking have opened up the possibility of gaze tracking during learning with ITSs and other technologies, thereby enabling a new generation of attention-aware cyberlearning.

6. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation (NSF) (DRL 1235958 and IIS 1523091). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

7. REFERENCES

[1] Arroyo, I. et al. 2007. Repairing disengagement with non-invasive interventions. *AIED* (2007), 195–202.

[2] Baker, R.S.J. 2007. Modeling and understanding students' off-task behavior in intelligent tutoring systems. *Proceedings of the SIGCHI conference on Human factors in computing systems* (2007), 1059–1068.

[3] Bixler, R. et al. 2015. Automatic Detection of Mind Wandering During Reading Using Gaze and Physiology. *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction* (New York, NY, USA, 2015), 299–306.

[4] Bixler, R. and D'Mello, S. 2015. Automatic Gaze-Based Detection of Mind Wandering with Metacognitive Awareness. *User Modeling, Adaptation and Personalization: 23rd International Conference, UMAP 2015, Dublin, Ireland, June 29 -- July 3, 2015. Proceedings*. F. Ricci et al., eds. Springer International Publishing. 31–43.

[5] Bixler, R. and D'Mello, S. 2015. Automatic gaze-based user-independent detection of mind wandering during computerized reading. *User Modeling and User-Adapted Interaction*. (2015), 1–36.

[6] Bixler, R. and D'Mello, S. 2014. Toward Fully Automated Person-Independent Detection of Mind Wandering. *User Modeling, Adaptation, and Personalization*. V. Dimitrova et al., eds. Springer International Publishing. 37–48.

[7] Blanchard, N. et al. 2014. Automated Physiological-Based Detection of Mind Wandering during Learning. *Intelligent Tutoring Systems*. S. Trausan-Matu et al., eds. Springer International Publishing. 55–60.

[8] Chawla, N.V. et al. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence research*. (2002), 321–357.

[9] Cocea, M. and Weibelzahl, S. 2011. Disengagement Detection in Online Learning: Validation Studies and Perspectives. *IEEE Transactions on Learning Technologies*. 4, 2 (Apr. 2011), 114–124.

[10] D'Mello, S.K. 2016. Giving Eyesight to the Blind: Towards Attention-Aware AIED. *International Journal of Artificial Intelligence in Education*. (2016), 1–15.

[11] Drummond, J. and Litman, D. 2010. In the Zone: Towards Detecting Student Zoning Out Using Supervised Machine Learning. *Intelligent Tutoring Systems*. V. Aleven et al., eds. Springer Berlin Heidelberg. 306–308.

[12] Franklin, M.S. et al. 2011. Catching the mind in flight: using behavioral indices to detect mindless reading in real time. *Psychon Bull Rev*. 18, 5 (Oct. 2011), 992–997.

[13] Freitas, A.A. 2002. *Data mining and knowledge discovery with evolutionary algorithms*.

[14] Hall, M. et al. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*. 11, 1 (2009), 10–18.

[15] Jeni, L.A. et al. 2013. Facing Imbalanced Data–Recommendations for the Use of Performance Metrics. *Proceedings of the 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction* (Washington, DC, USA, 2013), 245–251.

[16] Killingsworth, M.A. and Gilbert, D.T. 2010. A wandering mind is an unhappy mind. *Science*. 330, 6006 (2010), 932–932.

[17] Mills, C. et al. 2015. Mind Wandering During Learning with an Intelligent Tutoring System. *Artificial Intelligence in Education: 17th International Conference, AIED 2015, Madrid, Spain, June 22-26, 2015. Proceedings*. C. Conati et al., eds. Springer International Publishing. 267–276.

[18] Mills, C. et al. 2014. To Quit or Not to Quit: Predicting Future Behavioral Disengagement from Reading Patterns. *Intelligent Tutoring Systems* (2014), 19–28.

[19] Mills, C. and D'Mello, S. 2015. Toward a Real-time (Day) Dreamcatcher: Sensor-Free Detection of Mind Wandering During Online Reading. *Proceedings of the 8th International Conference on Educational Data Mining*. (2015).

[20] Mooneyham, B.W. and Schooler, J.W. 2013. The costs and benefits of mind-wandering: a review. *Can J Exp Psychol*. 67, 1 (Mar. 2013), 11–18.

[21] Olney, A.M. et al. In Press. Attention in Educational Contexts: The Role of the Learning Task in Guiding Attention. *The Handbook of Attention*. J. Fawcett et al., eds. MIT Press.

[22] Olney, A.M. et al. 2012. Guru: A Computer Tutor That Models Expert Human Tutors. *Intelligent Tutoring Systems*. S. Cerri et al., eds. Springer Berlin Heidelberg. 256–261.

[23] Powers, D.M.W. 2012. The Problem with Kappa. *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics* (Stroudsburg, PA, USA, 2012), 345–355.

[24] Randall, J.G. et al. 2014. Mind-wandering, cognition, and performance: a theory-driven meta-analysis of attention regulation. *Psychol Bull*. 140, 6 (Nov. 2014), 1411–1431.

[25] Rayner, K. 1998. Eye movements in reading and information processing: 20 years of research. *Psychol Bull*. 124, 3 (Nov. 1998), 372–422.

[26] Reichle, E.D. et al. 2010. Eye movements during mindless reading. *Psychol Sci*. 21, 9 (Sep. 2010), 1300–1310.

[27] Risko, E.F. et al. 2012. Everyday attention: variation in mind wandering and memory in a lecture. *Applied Cognitive Psychology*. 26, 2 (2012), 234–242.

[28] SharpNEAT: 2016. *http://sharpneat.sourceforge.net/*. Accessed: 2016-02-22.

[29] Smallwood, J. et al. 2007. Counting the cost of an absent mind: Mind wandering as an underrecognized influence on educational performance. *Psychonomic Bulletin & Review*. 14, 2 (2007), 230–236.

[30] Smallwood, J. 2011. Mind-wandering while reading: Attentional decoupling, mindless reading and the cascade model of inattention. *Language and Linguistics Compass*. 5, 2 (2011), 63–77.

[31] Smallwood, J. et al. 2008. When attention matters: the curious incident of the wandering mind. *Mem Cognit*. 36, 6 (Sep. 2008), 1144–1150.

[32] Smallwood, J. and Schooler, J.W. 2006. The restless mind. *Psychol Bull*. 132, 6 (Nov. 2006), 946–958.

[33] Smilek, D. et al. 2010. Out of mind, out of sight: eye blinking as indicator and embodiment of mind wandering. *Psychological Science*. 21, 6 (Jun. 2010), 786–789.

[34] Stanley, K.O. and Miikkulainen, R. 2002. Evolving Neural Networks Through Augmenting Topologies. *Evolutionary Computation*. 10, 2 (2002), 99–127.

[35] Vosskuhler, A. et al. 2008. OGAMA (Open Gaze and Mouse Analyzer): open-source software designed to analyze eye and mouse movements in slideshow study designs. *Behav Res Methods*. 40, 4 (Nov. 2008), 1150–1162.

[36] Wixon, M. et al. 2012. WTF? detecting students who are conducting inquiry without thinking fastidiously. *User Modeling, Adaptation, and Personalization*. Springer. 286–296.

How Deep is Knowledge Tracing?

Mohammad Khajah
Dept. of Computer Science
University of Colorado
Boulder, Colorado 80309
mohammad.khajah@colorado.edu

Robert V. Lindsey
Dept. of Computer Science
University of Colorado
Boulder, Colorado 80309
robert.lindsey@colorado.edu

Michael C. Mozer
Dept. of Computer Science
University of Colorado
Boulder, Colorado 80309
mozer@colorado.edu

ABSTRACT

In theoretical cognitive science, there is a tension between highly structured models whose parameters have a direct psychological interpretation and highly complex, general-purpose models whose parameters and representations are difficult to interpret. The former typically provide more insight into cognition but the latter often perform better. This tension has recently surfaced in the realm of educational data mining, where a deep learning approach to predicting students' performance as they work through a series of exercises—termed *deep knowledge tracing* or *DKT*—has demonstrated a stunning performance advantage over the mainstay of the field, *Bayesian knowledge tracing* or *BKT*. In this article, we attempt to understand the basis for DKT's advantage by considering the sources of statistical regularity in the data that DKT can leverage but which BKT cannot. We hypothesize four forms of regularity that BKT fails to exploit: recency effects, the contextualized trial sequence, inter-skill similarity, and individual variation in ability. We demonstrate that when BKT is extended to allow it more flexibility in modeling statistical regularities—using extensions previously proposed in the literature—BKT achieves a level of performance indistinguishable from that of DKT. We argue that while DKT is a powerful, useful, general-purpose framework for modeling student learning, its gains do not come from the discovery of novel representations—the fundamental advantage of deep learning. To answer the question posed in our title, knowledge tracing may be a domain that does *not* require 'depth'; shallow models like BKT can perform just as well and offer us greater interpretability and explanatory power.

1. INTRODUCTION

In the past forty years, machine learning and cognitive science have undergone many paradigm shifts, but few have been as dramatic as the recent surge of interest in *deep learning* [16]. Although deep learning is little more than a re-branding of neural network techniques popular around 1990, deep learning has achieved some remarkable results thanks to much faster computing resources and much larger data sets than were available in 1990. Deep learning underlies state-of-the-art systems in speech recognition, language processing, and image classification [16, 26]. Deep learning also is responsible for systems that can produce captions for images [29], create synthetic images [9], play video games [19] and even Go [27].

The 'deep' in deep learning refers to multiple levels of representation transformation that lie between model inputs and outputs. For example, an image-classification model may take pixel values as input and produce a labeling of the objects in the image as output. Between the input and output is a series of representation transformations that construct successively higher-order features—features that are less sensitive to lighting conditions and the position of objects in the image, and more sensitive to the identities of the objects and their qualitative relationships. The features discovered by deep learning exhibit a complexity and subtlety that make them difficult to analyze and understand (e.g., [31]). Furthermore, no human engineer could wire up a solution as thorough and accurate as solutions discovered by deep learning. Deep learning models are fundamentally *non-parametric*, in the sense that interpreting individual weights and individual unit activations in a network is pretty much impossible. This opacity is in stark contrast to parametric models, e.g., linear regression, where each of the coefficients has a clear interpretation in terms of the problem at hand and the input features.

In one domain after the next, deep learning has achieved gains over traditional approaches. Deep learning discards hand-crafted features in favor of representation learning, and deep learning often ignores domain knowledge and structure in favor of massive data sets and general architectural constraints on models (e.g., models with spatial locality to process images, and models with local temporal constraints to process time series).

It was inevitable that deep learning would be applied to student-learning data [22]. This domain has traditionally been the purview of the educational data mining community, where *Bayesian knowledge tracing*, or *BKT*, is the dominant computational approach [3]. The deep learning approach to modeling student data, termed *deep knowledge tracing* or *DKT*, created a buzz when it appeared at the Neural Information Processing Systems Conference in December 2015, including press inquiries (N. Heffernan, personal communi-

cation) and descriptions of the work in the blogosphere (e.g., [7]). Piech et al. [22] reported substantial improvements in prediction performance with DKT over BKT on two real-world data sets (ASSISTMENTS, KHAN ACADEMY) and one synthetic data set which was generated under assumptions that are not tailored to either DKT or BKT. DKT achieves a reported 25% gain in AUC (a measure of prediction quality) over the best previous result on the ASSISTMENTS benchmark.

In this article, we explore the success of DKT. One approach to this exploration might be to experiment with DKT, removing components of the model or modifying the input data to determine which model components and data characteristics are essential to DKT's performance. We pursue an alternative approach in which we first formulate hypotheses concerning the signals in the data that DKT is able to exploit but that BKT is not. Given these hypotheses, we propose extensions to BKT which provide it with additional flexibility, and we evaluate whether the enhanced BKT can achieve results comparable to DKT. This procedure leads not only to a better understanding of how BKT and DKT differ, but also helps us to understand the structure and statistical regularities in the data source.

1.1 Modeling Student Learning

The domain we're concerned with is electronic tutoring systems which employ cognitive models to track and assess student knowledge. Beliefs about what a student knows and doesn't know allow a tutoring system to dynamically adapt its feedback and instruction to optimize the depth and efficiency of learning.

Ultimately, the measure of learning is how well students are able to apply skills that they have been taught. Consequently, student modeling is often formulated as time series prediction: given the series of exercises a student has attempted previously and the student's success or failure on each exercise, predict how the student will fare on a new exercise. Formally, the data consist of a set of binary random variables indicating whether student s produces a correct response on trial t, $\{X_{st}\}$. The data also include the exercise labels, $\{Y_{st}\}$, which characterize the exercise. Secondary data has also been incorporated in models, including the student's utilization of hints, response time, and characteristics of the specific exercise and the student's particular history with related exercises [2, 30]. Although such data improve predictions, the bulk of research in this area has focused on the primary measure—whether a response is correct or incorrect—and a sensible research strategy is to determine the best model based on the primary data, and then to determine how to incorporate secondary data.

The exercise label, Y_{st}, might index the specific exercise, e.g., $3 + 4$ versus $2 + 6$, or it might provide a more general characterization of the exercise, e.g., *single digit addition*. In the latter case, exercise are grouped by the *skill* that must be applied to obtain a solution. Although we will use the term skill in this article, others refer to the skill as a *knowledge component*, and the authors of DKT also use the term *concept*. Regardless, the important distinction for the purpose of our work is between a label that indicates the particular exercise and a label that indicates the general skill required to perform the exercise. We refer to these two types of labels as *exercise indexed* and *skill indexed*, respectively.

1.2 Knowledge Tracing

BKT models skill-specific performance, i.e., performance on a series of exercises that all tap the same skill. A separate instantiation of BKT is made for each skill, and a student's raw trial sequence is parsed into skill-specific subsequences that preserve the relative ordering of exercises within a skill but discard the ordering relationship of exercises across skills. For a given skill σ, BKT is trained using the data from each student s, $\{X_{st}|Y_{st} = \sigma\}$, where the relative trial order is preserved. Because it will become important for us to distinguish between absolute trial index and the relative trial index within a skill, we use t to denote the former and use i to denote the latter.

BKT is based on a theory of all-or-none human learning [1] which postulates that the knowledge state of student s following the i'th exercise requiring a certain skill, K_{si}, is binary: 1 if the skill has been mastered, 0 otherwise. BKT, formalized as a hidden Markov model, infers K_{si} from the sequence of observed responses on trials $1 \ldots i$, $\{X_{s1}, X_{s2}, \ldots, X_{si}\}$. BKT is typically specified by four parameters: $P(K_{s0} = 1)$, the probability that the student has mastered the skill prior to solving the first exercise; $P(K_{s,i+1} = 1 \mid K_{si} = 0)$, the transition probability from the not-mastered to mastered state; $P(X_{si} = 1 \mid K_{si} = 0)$, the probability of correctly *guessing* the answer prior to skill mastery; and $P(X_{si} = 0 \mid K_{si} = 1)$, the probability of answering incorrectly due to a *slip* following skill mastery. Because BKT is typically used in modeling practice over brief intervals, the model assumes no forgetting, i.e., K cannot transition from 1 to 0.

BKT is a highly constrained, structured model. It assumes that the student's knowledge state is binary, that predicting performance on an exercise requiring a given skill depends only on the student's binary knowledge state, and that the skill associated with each exercise is known in advance. If correct, these assumptions allow the model to make strong inferences. If incorrect, they limit the model's performance. The only way to determine if model assumptions are correct is to construct an alternative model that makes different assumptions and to determine whether the alternative outperforms BKT. DKT is exactly this alternative model, and its strong performance directs us to examine BKT's limitations. First, however, we briefly describe DKT.

Rather than constructing a separate model for each skill, DKT models all skills jointly. The input to the model is the complete sequence of exercise-performance pairs, $\{(X_{s1}, Y_{s1}) \ldots (X_{st}, Y_{st}) \ldots (X_{sT}, Y_{sT})\}$, presented one trial at a time. As depicted in Figure 1, DKT is a recurrent neural net which takes (X_{st}, Y_{st}) as input and predicts $X_{s,t+1}$ for each possible exercise label. The model is trained and evaluated based on the match between the actual and predicted $X_{s,t+1}$ for the tested exercise $(Y_{s,t+1})$. In addition to the input and output layers representing the current trial and the next trial, respectively, the network has a hidden layer with fully recurrent connections (i.e., each hidden unit connects back to all other hidden units). The hidden layer thus serves to retain relevant aspects of the input history as they are use-

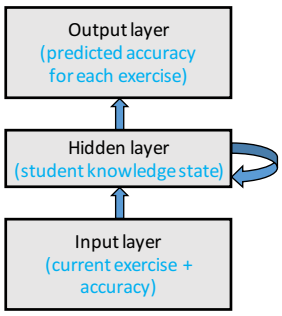


Figure 1: Deep knowledge tracing (DKT) architecture. Each rectangle depicts a set of processing units; each arrow depicts complete connectivity between each unit in the source layer and each unit in the destination layer.

ful for predicting future performance. The hidden state of the network can be conceived of as embodying the student's knowledge state. Piech et al. [22] used a particular type of hidden unit, called an LSTM (long short-term memory) [10], which is interesting because these hidden units behave very much like the BKT latent knowledge state, K_{si}. To briefly explain LSTM, each hidden unit acts like a memory element that can hold a bit of information. The unit is triggered to turn on or off by events in the input or the state of other hidden units, but when there is no specific trigger, the unit preserves its state, very similar to the way that the latent state in BKT is sticky—once a skill is learned it stays learned. With 200 LSTM hidden units—the number used in simulations reported in [22]—and 50 skills, DKT has roughly 250,000 free parameters (connection strengths). Contrast this number with the 200 free parameters required for embodying 50 different skills in BKT.

With its thousand-fold increase in flexibility, DKT is a very general architecture. One can implement BKT-like dynamics in DKT with a particular, restricted set of connection strengths. However, DKT clearly has the capacity to encode learning dynamics that are outside the scope of BKT. This capacity is what allows DKT to discover structure in the data that BKT misses.

1.3 Where Does BKT Fall Short?

In this section, we describe four regularities that we conjecture to be present in the student-performance data. DKT is flexible enough that it has the potential to discover these regularities, but the more constrained BKT model is simply not crafted to exploit the regularities. In following sections, we suggest means of extending BKT to exploit such regularities, and conduct simulation studies to determine whether the enhanced BKT achieves performance comparable to that of DKT.

1.3.1 Recency Effects

Human behavior is strongly recency driven. For example, when individuals perform a choice task repeatedly, response latency can be predicted by an exponentially decaying average of recent stimuli [12]. Intuitively, one might expect to observe recency effects in student performance. Consider, for example, a student's time varying engagement. If the level of engagement varies slowly relative to the rate at which exercises are being solved, a correlation would be induced in performance across local spans of time. A student who performed poorly on the last trial because they were distracted is likely to perform poorly on the current trial. We conducted a simple assessment of recency using the ASSISTMENTS data set (the details of this data set will be described shortly). Similarly to [5], we built an autoregressive model that predicts performance on the current trial as an exponentially weighted average of performance on past trials, with a decay half life of about 5 steps. We found that this single parameter model fit the ASSISTMENTS data reliably better than classic BKT. (We are not presenting details of this simulation because we will evaluate a more rigorous variant of the idea in a following section. Our goal here is to convince the reader that there is likely some value to the notion of recency-weighted prediction.)

Recurrent neural networks tend to be more strongly influenced by recent events in a sequence than more distal events [20]. Consequently, DKT is well suited to exploiting recent performance in making predictions. In contrast, the generative model underlying BKT supposes that once a skill is learned, performance will remain strong, and that a slip at time t is independent of a slip at $t+1$.

1.3.2 Contextualized Trial Sequence

The psychological literature on practice of multiple skills indicates that the sequence in which an exercise is embedded influences learning and retention (e.g., [24, 25]). For example, given three exercises each of skills A and B, presenting the exercises in the *interleaved* order A_1–B_1–A_2–B_2–A_3–B_3 yields superior performance relative to presenting the exercises in the *blocked* order A_1–A_2–A_3–B_1–B_2–B_3. (Performance in this situation can be based on an immediate or delayed test.)

Because DKT is fed the entire sequence of exercises a student receives in the order the student receives them, it can potentially infer the effect of exercise order on learning. In contrast, because classic BKT separates exercises by skill, preserving only the relative order of exercises within a skill, the training sequence for BKT is the same regardless of whether the trial order is blocked or interleaved.

1.3.3 Inter-Skill Similarity

Each exercise presented to a student has an associated label. In typical applications of BKT—as well as two of the three simulations reported in Piech et al. [22]—the label indicates the skill required to solve the problem. Any two such skills, S_1 and S_2, may vary in their degree of relatedness. The stronger the relatedness, the more highly correlated one would expect performance to be on exercises tapping the two skills, and the more likely that the two skills will be learned simultaneously.

DKT has the capacity to encode inter-skill similarity. If each hidden unit represents student knowledge state for a particular skill, then the hidden-to-hidden connections encode the degree of overlap. In an extreme case, if two skills are highly similar, they can be modeled by a single hidden knowledge state. In contrast, classic BKT treats each skill as an in-

dependent modeling problem and thus can not discover or leverage inter-skill similarity.

DKT has the additional strength, as demonstrated by Piech et al., that it can accommodate the absence of skill labels. If each label simply indexes a specific exercise, DKT can discover interdependence between exercises in exactly the same manner as it discovers interdependence between skills. In contrast, BKT requires exercise labels to be skill indexed.

1.3.4 Individual Variation in Ability

Students vary in ability, as reflected in individual differences in mean accuracy across trials and skills. Individual variation might potentially be used in a predictive manner: a student's accuracy on early trials in a sequence might predict accuracy on later trials, regardless of the skills required to solve exercises. We performed a simple verification of this hypothesis using the ASSISTMENTS data set. In this data set, students study one skill at a time and then move on to the next skill. We computed correlation between mean accuracy of all trials on the first n skills and the mean accuracy of all trials on skill $n+1$, for all students and for $n \in \{1, ..., N-1\}$ where N is the number of skills a student studied. We obtained a correlation coefficient of 0.39: students who tend to do well on the early skills learned tend to do well on later skills, regardless of the skills involved.

DKT is presented with a student's complete trial sequence. It can use a student's average accuracy up to trial t to predict trial $t+1$. Because BKT models each skill separately from the others, it does not have the contextual information needed to estimate a student's average accuracy or overall ability.

2. EXTENDING BKT

In the previous section, we described four regularities that appear to be present in the data and which we conjecture that DKT exploits but which the classic BKT model cannot. In this section, we describe three extensions to BKT that would bring BKT on par with DKT with regard to these regularities.

2.1 Forgetting

To better capture recency effects, BKT can be augmented to allow for forgetting of skills. Forgetting corresponds to fitting a BKT parameter $F \equiv P(K_{s,i+1} = 0 \mid K_{si} = 1)$, the probability of transitioning from a state of knowing to not knowing a skill. In standard BKT, $F = 0$.

Without forgetting, once BKT infers that the student has learned, even a long run of poorly performing trials cannot alter the inferred knowledge state. However, with forgetting, the knowledge state can transition in either direction, which allows the model to be more sensitive to the recent trials: A run of unsuccessful trials is indicative of not knowing the skill, regardless of what preceded the run. Forgetting is not a new idea to BKT, and in fact was included in the original psychological theory that underlies the notion of binary knowledge state [1]. However, it has not typically been incorporated into BKT. When it has been included in BKT [23], the motivation was to model forgetting from one day to the next, not forgetting that can occur on a much shorter time scale.

Incorporating forgetting can not only sensitize BKT to recent events but can also contextualize trial sequences. To explain, consider an exercise sequence such as A_1–A_2–B_1–A_3–B_2–B_3–A_4, where the labels are instances of skills A and B. Ordinary BKT discards the absolute number of trials between two exercises of a given skill, but with forgetting, we can count the number of intervening trials and treat each as an independent opportunity for forgetting to occur. Consequently, the probability of forgetting between A_1 and A_2 is F, but the probability of forgetting between A_2 and A_3 is $1-(1-F)^2$ and between A_3 and A_4 is $1-(1-F)^3$. Using forgetting, BKT can readily incorporate some information about the absolute trial sequence, and therefore has more potential than classic BKT to be sensitive to interspersed trials in the exercise sequence.

2.2 Skill Discovery

To model interactions among skills, one might suppose that each skill has some degree of influence on the learning of other skills, not unlike the connection among hidden units in DKT. For BKT to allow for such interactions among skills, the independent BKT models would need to be interconnected, using an architecture such as a factorial hidden Markov model [6]. As an alternative to this somewhat complex approach, we explored a simpler scheme in which different exercise labels could be collapsed together to form a single skill. For example, consider an exercise sequence such as A_1–B_1–A_2–C_1–B_2–C_2–C_3. If skills A and B are highly similar or overlapping, such that learning one predicts learning the other, it would be more sensible to treat this sequence in a manner that groups A and B into a single skill, and to train a single BKT instantiation on both A and B trials. This approach can be used whether the exercise labels are skill indices or exercise indices. (One of the data sets used by Piech et al. [22] to motivate DKT has exercise-indexed labels).

We recently proposed an inference procedure that automatically discovers the cognitive skills needed to accurately model a given data set [18]. (A related procedure was independently proposed in [8].) The approach couples BKT with a technique that searches over partitions of the exercise labels to simultaneously (1) determine which skill is required to correctly answer each exercise, and (2) model a student's dynamical knowledge state for each skill. Formally, the technique assigns each exercise label to a latent skill such that a student's expected accuracy on a sequence of same-skill exercises improves monotonically with practice according to BKT. Rather than discarding the skills identified by experts, our technique incorporates a nonparametric prior over the exercise-skill assignments that is based on the expert-provided skills and a weighted Chinese restaurant process [11].

In the above illustration, our technique would group A and B into one skill and C into another. This procedure collapses like skills (or like exercises), yielding better fits to the data by BKT. Thus, the procedure performs a sort of *skill discovery*.

2.3 Incorporating Latent Student-Abilities

To account for individual variation in student ability, we have extended BKT [14, 13] such that slip and guess prob-

abilities are modulated by a latent *ability* parameter that is inferred from the data, much in the spirit of item-response theory [4]. As we did in [14], we assume that students with stronger abilities have lower slip and higher guess probabilities. When the model is presented with new students, the posterior predictive distribution on abilities is used initially, but as responses from the new student are observed, uncertainty in the student's ability diminishes, yielding better predictions for the student.

3. SIMULATIONS

3.1 Data Sets

Piech et al. [22] studied three data sets. One of the data sets, from Khan Academy, is not publicly available. Despite our requests and a plea from one of the co-authors of the DKT paper, we were unable to obtain permission from the data science team at Khan Academy to use the data set. We did investigate the other two data sets in Piech et al., which are as follows.

ASSISTMENTS is an electronic tutor that teaches and evaluates students in grade-school math. The 2009-2010 "skill builder" data set is a large, standard benchmark, available by searching the web for *assistment-2009-2010-data*. We used the train/test split provided by Piech et al., and following Piech et al., we discarded all students who had only a single trial of data.

SYNTHETIC is a synthetic data set created by Piech et al. to model virtual students learning virtual skills. The training and test sets each consist of 2000 virtual students performing the same sequence of 50 exercises drawn from 5 skills. The exercise on trial t is assumed to have a difficulty characterized by δ_t and require a skill specified by σ_t. The exercises are labeled by the *identity of the exercise*, not by the underlying skill, σ_t. The ability of a student, denoted, α_t varies over time according to a drift-diffusion process, generally increasing with practice. The response correctness on trial t is a Bernoulli draw with probability specified by guessing-corrected item-response theory with difficulty and ability parameters δ_t and α_t. This data set is challenging for BKT because the skill assignments, σ_t, are not provided and must be inferred from the data. Without the skill assignments, BKT must be used either with all exercises associated with a single skill or each exercise associated with its own skill. Either of these assumptions will miss important structure in the data. SYNTHETIC is an interesting data set in that the underlying generative model is neither a perfect match to DKT or BKT (even with the enhancements we have described). The generative model seems realistic in its assumption that knowledge state varies continuously.

We included two additional data sets in our simulations. SPANISH is a data set of 182 middle-school students practicing 409 Spanish exercises (translations and application of simple skills such as verb conjugation) over the course of a 15-week semester, with a total of 578,726 trials [17]. STATICS is from a college-level engineering statics course with 189,297 trials and 333 students and 1,223 exercises [28], available from the PSLC DataShop web site [15].

3.2 Methods

We evaluated five variants of BKT[1], each of which incorporates a different subset of the extensions described in the previous section: a base version that corresponds to the classic model and the model against which DKT was evaluated in [22], which we'll refer to simply as BKT; a version that incorporates forgetting ($BKT{+}F$), a version that incorporates skill discovery ($BKT{+}S$), a version that incorporates latent abilities ($BKT{+}A$), and a version that incorporates all three of the extensions ($BKT{+}FSA$). We also built our own implementation of DKT with LSTM recurrent units[2]. (Piech et al. described the LSTM version as better performing, but posted only the code for the standard recurrent neural net version.) We verified that our implementation produced results comparable to those reported in [22] on ASSISTMENTS and SYNTHETIC. We then also ran the model on SPANISH and STATICS.

For ASSISTMENTS, SPANISH, and STATICS, we used a single train/test split. The ASSISTMENTS train/test split was identical to that used by Piech et al. For SYNTHETIC, we used the 20 simulation sets provided by Piech et al. and averaged results across the 20 simulations.

Each model was evaluated on each domain's test data set, and the performance of the model was quantified with a discriminability score, the *area under the ROC curve* or *AUC*. AUC is a measure ranging from .5, reflecting no ability to discriminate correct from incorrect responses, to 1.0, reflecting perfect discrimination. AUC is computed by obtaining a prediction on the test set for each trial, across all skills, and then using the complete set of predictions to form the ROC curve. Although Piech et al. [22] do not describe the procedure they use to compute AUC for DKT, code they have made available implements the procedure we describe, and not the obvious alternative procedure in which ROC curves are computed on a per-skill basis and then averaged to obtain an overall AUC.

3.3 Results

Figure 2 presents the results of our comparison of five variants of BKT on the four data sets. We walk through the data sets from left to right.

On ASSISTMENTS, classic BKT obtains an AUC of 0.73, better than the 0.67 reported for BKT by Piech et al. We are not sure why the scores do not match, although 0.67 is close to the AUC score we obtain if we treat all exercises as associated with a single skill or if we compute AUC on a per-skill basis and then average.[3] BKT+F obtains an AUC of 0.83,

[1]https://github.com/robert-lindsey/WCRP/tree/forgetting
[2]https://github.com/mmkhajah/dkt
[3]Piech et al. cite Pardos and Heffernan [21] as obtaining BKT's best reported performance on ASSISTMENTS—an AUC of 0.69. In [21], the overall AUC is computed by averaging the per-skill AUCs. This method yields a lower score than the method used by Piech et al., for two reasons. First, the Piech procedure weighs all *trials* equally, whereas the Pardos and Heffernan procedure weighs all *skills* equally. With the latter procedure, the overall AUC will be dinged if the model does poorly on a skill with just a few trials, as we have observed to be the case with ASSISTMENTS. The latter procedure also produces a lower overall AUC because it suppresses any lift due to being able to predict the relative accuracy of different skills. In summary, it appears that

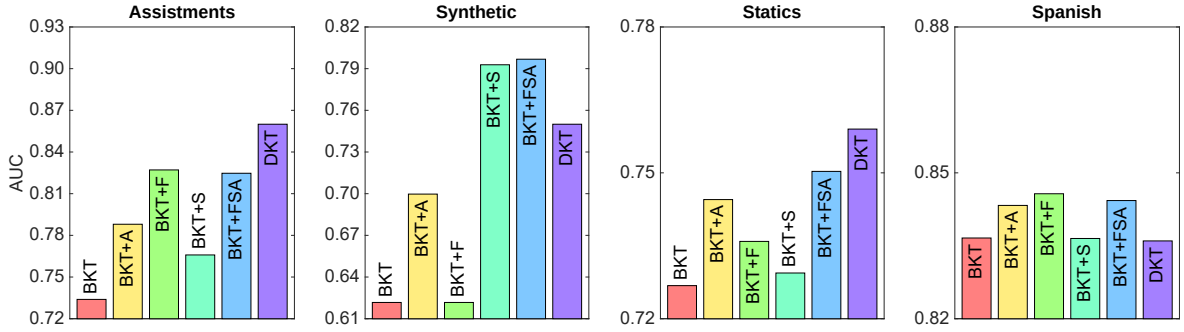


Figure 2: A comparison of six models on four data sets. Model performance on the test set is quantified by AUC, a measure of how well the model discriminates (predicts) correct and incorrect student responses. The models are trained on one set of students and tested on another set. Note that the AUC scale is different for each graph, but tic marks are always spaced by .03 units in AUC. On ASSISTMENTS and SYNTHETIC, DKT results are from Piech et al. [22]; on STATICS and SPANISH DKT results are from our own implementation. BKT= classic Bayesian knowledge tracing; BKT+A= BKT with inference of latent student abilities; BKT+F= BKT with forgetting; BKT+S= BKT with skill discovery; BKT+FSA= BKT with all three extensions; DKT= deep knowledge tracing

not quite as good as the 0.86 value reported for DKT by Piech et al. Examining the various enhancements to BKT, AUC is boosted both by incorporating forgetting and by incorporating latent student abilities. We find it somewhat puzzling that the combination of the two enhancements, embodied in BKT+FSA, does no better than BKT+F or BKT+A, considering that the two enhancements tap different properties of the data: the student abilities help predict transfer from one skill to the next, whereas forgetting facilitates prediction within a skill.

To summarize the comparison of BKT and DKT, 31.6% of difference in performance reported in [22] appears to be due to the use of a biased procedure for computing the AUC for BTK. Another 50.6% of the difference in performance reported vanishes if BKT is augmented to allow for forgetting. We can further improve BKT if we allow the skill discovery algorithm to operate with exercise labels that index individual exercises, as opposed to labels that index the skill associated with each exercise. With exercise-indexed labels, BKT+S and BKT+FSA both obtain an AUC of 0.90, beating DKT. However, given DKT's ability to perform skill discovery, we would not be surprised if it also achieved a similar level of performance when allowed to exploit exercise-indexed labels.

Turning to SYNTHETIC, classic BKT obtains an AUC of 0.62, again significantly better than the 0.54 reported by Piech et al. In our simulation, we treat each exercise as having a distinct skill label, and thus BKT learns nothing more than the mean performance level for a specific exercise. (Because the exercises are presented in a fixed order, the exercise identity and the trial number are confounded. Because performance tends to improve as trials advance in the synthetic data, BKT is able to learn this relationship.) It is possible here that Piech et al. treated all exercises as associated with a single skill or that they used the biased procedure for computing AUC; either of these explanations is consistent with their reported AUC of 0.54.

Regarding the enhancements to BKT, adding student abilities (BKT+A) improves prediction of SYNTHETIC which is understandable given that the generative process simulates students with abilities that vary slowly over time. Adding forgetting (BKT+F) does not help, consistent with the generative process which assumes that knowledge level is on average increasing with practice; there is no systematic forgetting in the student simulation. Critical to this simulation is skill induction: BKT+S and BKT+FSA achieve an AUC of 0.80, better than the reported 0.75 for DKT in [22].

On STATICS, each BKT extension obtains an improvement over classic BKT, although the magnitude of the improvements are small. The full model, BKT+FSA, obtains an AUC of 0.75 and our implementation of DKT obtains a nearly identical AUC of 0.76. On SPANISH, the BKT extensions obtain very little benefit. The full model, BKT+FSA, obtains an AUC of 0.846 and again, DKT obtains a nearly identical AUC of 0.836. These two sets of results indicate that for at least some data sets, classic BKT has no glaring deficiencies. However, we note that BKT model accuracy can be improved if algorithms are considered that use exercise labels which are indexed by exercise and not by skill. For example, with STATICS, performing skill discovery using exercise-indexed labels, [17] obtain an AUC of 0.81, much better than the score of 0.73 we report here for BKT+S based on skill-indexed labels.

In summary, enhanced BKT appears to perform as well on average as DKT across the four data sets. Enhanced BKT outperforms DKT by 20.0% (.05 AUC units) on SYNTHETIC and by 3.0% (.01 AUC unit) on SPANISH. Enhanced BKT underperforms DKT by 8.3% (.03 AUC units) on ASSISTMENTS and by 3.5% (.01 AUC unit) on STATICS. These percentages are based on the difference of AUCs scaled by by $\text{AUC}_{\text{DKT}} - 0.5$, which takes into account the fact that an AUC of 0.5 indicates no discriminability.

inconsistent procedures may have been used to compute performance of BKT versus DKT in [22], and the procedure for BKT is biased to yield a lower score.

4. DISCUSSION

Our goal in this article was to investigate the basis for the impressive predictive advantage of deep knowledge tracing over Bayesian knowledge tracing. We found some evidence that different procedures may have been used to evaluate DKT and BKT in [22], leading to a bias against BKT. When we replicated simulations of BKT reported in [22], we obtained significantly better performance: an AUC of 0.73 versus 0.67 on ASSISTMENTS, and an AUC of 0.62 versus 0.54 on SYNTHETIC.

However, even when the bias is eliminated, DKT obtains real performance gains over BKT. To understand the basis for these gains, we hypothesized various forms of regularity in the data which BKT is not able to exploit. We proposed enhancements to BKT to allow it to exploit these regularities, and we found that the enhanced BKT achieved a level of performance on average indistinguishable from that of DKT over the four data sets tested. The enhancements we explored are not novel; they have previously been proposed and evaluated in the literature. They include forgetting [23], latent student abilities [14, 13, 21], and skill induction [17, 8].

We observe that different enhancements to BKT matter for different data sets. For ASSISTMENTS, incorporating forgetting is key; forgetting allows BKT to capture recency effects. For SYNTHETIC, incorporating skill discovery yielded huge gains, as one would expect when the exercise-skill mapping is not known. And for STATICS, incorporating latent student abilities was relatively most beneficial; these abilities enable the model to tease apart the capability of a student and the intrinsic difficulty of an exercise or skill. Of the three enhancements, forgetting and student abilities are computationally inexpensive to implement, whereas skill discovery adds an extra layer of computational complexity to inference.

The elegance of DKT is apparent when one considers the effort we have invested to bring BKT to par with DKT. DKT did not require its creators to analyze the domain and determine sources of structure in the data. In contrast, our approach to augmenting BKT required some domain expertise, a thoughtful analysis of BKT's limitations, and distinct solutions to each limitation. DKT is a generic recurrent neural network model [10], and it has no constructs that are specialized to modeling learning and forgetting, discovering skills, or inferring student abilities. This flexibility makes DKT robust on a variety of datasets with little prior analysis of the domains. Although training recurrent networks is computationally intensive, tools exist to exploit the parallel processing power in graphics processing units (GPUs), which means that DKT can scale to large datasets. Classic BKT is inexpensive to fit, although the variants we evaluated—particularly the model that incorporates skill discovery—require computation-intensive MCMC methods that have a distinct set of issues when it comes to parallelization.

DKT's advantages come at a price: interpretability. DKT is massive neural network model with tens of thousands of parameters which are near-impossible to interpret. Although the creators of DKT did not have to invest much up-front time analyzing their domain, they did have to invest substantive effort to understand what the model had actually learned. Our proposed BKT extensions achieve predictive performance similar to DKT whilst remaining interpretable: the model parameters (forgetting rate, student ability, etc.) are psychologically meaningful. When skill discovery is incorporated into BKT, the result is clear: a partition of exercises into skills. Reading out such a partitioning from DKT is challenging and only an approximate representation of the knowledge in DKT.

Finally, we return to the question posed in the paper's title: How deep is knowledge tracing? Deep learning refers to the discovery of representations. Our results suggest that representation discovery is not at the core of DKT's success. We base this argument on the fact that our enhancements to BKT bring it to the performance level of DKT *without* requiring any sort of subsymbolic representation discovery.[4] Representation discovery is clearly critical in perceptual domains such as image or speech classification. But the domain of education and student learning is high level and abstract. The input and output elements of models are psychologically meaningful. The relevant internal states of the learner have some psychological basis. The characterization of exercises and skills can—to at least a partial extent—be expressed symbolically.

Instead of attributing DKT's success to representation discovery, we attribute DKT's success to its flexibility and generality in capturing statistical regularities directly present in the inputs and outputs. As long as there are sufficient data to constrain the model, DKT is more powerful than classic BKT. BKT arose in a simpler era, an era in which data and computation resources were precious. DKT reveals the value of relaxing these constraints in the big data era. But despite the wild popularity of deep learning, there are many ways to relax the constraints and build more powerful models other than creating a black box predictive device with a vast interconnected tangle of connections and parameters that are nearly impossible to interpret.

5. ACKNOWLEDGMENTS

This research was supported by NSF grants SES-1461535, SBE-0542013, and SMA-1041755.

6. REFERENCES

[1] R. Atkinson and J. A. Paulson. An approach to the psychology of instruction. *Psychology Bulletin*, 78:49–61, 1972.

[2] R. S. Baker, A. T. Corbett, and V. Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian knowledge tracing. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, pages 406–415, Berlin, Heidelberg, 2008. Springer-Verlag.

[3] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modelling the acquisition of procedural knowledge. *User Model. User-Adapt. Interact.*, 4(4):253–278, 1995.

[4] Of course, the skill discovery mechanism we incorporated certainly does regroup exercises to form skills, but the form of this regrouping or partitioning is far more limited than the typical transformations in a neural network to map from one level of representation to another.

[4] P. De Boeck and M. Wilson. *Explanatory Item Response Models: a Generalized Linear and Nonlinear Approach*. Springer-Verlag, New York, NY, 2004.

[5] A. Galyardt and I. Goldin. Move your lamp post: Recent data reflects learner knowledge better than older data. *JEDM-Journal of Educational Data Mining*, 7(2):83–108, 2015.

[6] Z. Ghahramani and M. I. Jordan. Factorial hidden markov models. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 472–478. MIT Press, 1996.

[7] R. Golden. How to optimize student learning using recurrent neural networks (educational technology). Web page, 2016. http://tinyurl.com/GoldenDKT, retrieved February 29, 2016.

[8] J. P. Gonzales-Brenes. Modeling skill acquisition over time with sequence and topic modeling. In S. V. N. V. G. Lebanon, editor, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*. JMLR, 2015.

[9] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. DRAW: A recurrent neural network for image generation. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1462–1471, 2015.

[10] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] H. Ishwaran and L. F. James. Generalized weighted chinese restaurant processes for species sampling mixture models. *Statistica Sinica*, pages 1211–1235, 2003.

[12] M. Jones, T. Curran, M. C. Mozer, and M. H. Wilder. Sequential effects in response time reveal learning mechanisms and event representations. *Psychological review*, 120:628–666, 2013.

[13] M. Khajah, Y. Huang, J. P. Gonzales-Brenes, M. C. Mozer, and P. Brusilovsky. Integrating knowledge tracing and item response theory: A tale of two frameworks. In M. Kravcik, O. C. Santos, and J. G. Boticario, editors, *Proceedings of the 4th International Workshop on Personalization Approaches in Learning Environments*, pages 7–15. CEUR Workshop Proceedings, 2014.

[14] M. Khajah, R. M. Wing, R. V. Lindsey, and M. C. Mozer. Incorporating latent factors into knowledge tracing to predict individual differences in learning. In J. Stamper, Z. Pardos, M. Mavrikis, and B. M. McLaren, editors, *Proceedings of the 7th International Conference on Educational Data Mining*, pages 99–106. Educational Data Mining Society Press, 2014.

[15] K. Koedinger, R. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the EDM community: The PSLC DataShop. In C. Romero, S. Ventura, M. Pechenizkiy, and R. Baker, editors, *Handbook of Educ. Data Mining*, http://pslcdatashop.org, 2010.

[16] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[17] R. Lindsey, J. Shroyer, H. Pashler, and M. Mozer. Improving student's long-term knowledge retention with personalized review. *Psychological Science*, 25:639–47, 2014.

[18] R. V. Lindsey, M. Khajah, and M. C. Mozer. Automatic discovery of cognitive skills to improve the prediction of student learning. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1386–1394. Curran Associates, Inc., 2014.

[19] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[20] M. C. Mozer. Induction of multiscale temporal structure. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 275–282. Morgan-Kaufmann, 1992.

[21] Z. A. Pardos and N. T. Heffernan. KT-IDEM: Introducing item difficulty to the knowledge tracing model. In *User Modeling, Adaption and Personalization*, pages 243–254. Springer, 2011.

[22] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 505–513. Curran Associates, Inc., 2015.

[23] Y. Qiu, Y. Qi, H. Lu, Z. A. Pardos, and N. T. Heffernan. Does time matter? modeling the effect of time with Bayesian knowledge tracing. In M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero, and J. C. Stamper, editors, *Educational Data Mining 2011*, pages 139–148. www.educationaldatamining.org, 2011.

[24] D. Rohrer, R. F. Dedrick, and K. Burgess. The benefit of interleaved mathematics practice is not limited to superficially similar kinds of problems. *Psychonomic Bulletin and Review*, 21:1323–1330, 2014.

[25] D. Rohrer, R. F. Dedrick, and S. Stershic. Interleaved practice improves mathematics learning. *Journal of Educational Psychology*, 107:900–908, 2015.

[26] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[27] D. Sliver, A. Huang, C. J. Maddison, A. Guez, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[28] P. Steif and N. Bier. OLI Engineering Statics – Fall 2011. Feb. 2014.

[29] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition*, 2015.

[30] H.-F. Yu and Others. Feature engineering and classifier ensemble for KDD cup 2010. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2010.

[31] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, pages 818–833. Springer International Publishing, Cham, 2014.

Temporally Coherent Clustering of Student Data

Severin Klingler
Department of Computer Science
ETH Zurich, Switzerland
kseverin@inf.ethz.ch

Tanja Käser
Department of Computer Science
ETH Zurich, Switzerland
kaesert@inf.ethz.ch

Barbara Solenthaler
Department of Computer Science
ETH Zurich, Switzerland
sobarbar@inf.ethz.ch

Markus Gross
Department of Computer Science
ETH Zurich, Switzerland
grossm@inf.ethz.ch

ABSTRACT

The extraction of student behavior is an important task in educational data mining. A common approach to detect similar behavior patterns is to cluster sequential data. Standard approaches identify clusters at each time step separately and typically show low performance for data that inherently suffer from noise, resulting in temporally inconsistent clusters. We propose an evolutionary clustering pipeline that can be applied to learning data, aiming at improving cluster stability over multiple training sessions in the presence of noise. Our model selection is designed such that relevant cluster evolution effects can be captured. The pipeline can be used as a black box for any intelligent tutoring system (ITS). We show that our method outperforms previous work regarding clustering performance and stability on synthetic data. Using log data from two ITS, we demonstrate that the proposed pipeline is able to detect interesting student behavior and properties of learning environments.

Keywords

Evolutionary Clustering, Markov Chains, Sequence Mining, Distance Metrics

1. INTRODUCTION

The extraction of student properties is a central element in educational data mining. On the one hand, the identification of student abilities and behavior patterns allows us to draw conclusions about human learning. On the other hand, the extracted properties can be used to improve the adaptation of the underlying intelligent tutoring system (ITS).

Clustering of sequential data is a common approach to detect similar behavior patterns and has been successfully applied to a variety of applications such as reading comprehension [22], online collaboration tools [24], table-top environments [19], web browsing [25], physics simulations [4] or homework assignments [11]. Furthermore, a variety of different student behavior has been investigated. [20] identified students that impose challenges for the student models. Other work studied the relation between interaction patterns and the performance of students [3, 14] and the relation between student action sequences and their affective states [3].

Common techniques for the analysis of sequential data include sequence mining [1, 19], differential pattern mining [11] or Hidden Markov models (HMM) [5, 6]. Sequential pattern mining techniques have been contextualized using piecewise linear segmentation [14]. Others have employed semi-supervised graph clustering using the predictions from a student model as additional constraints [20]. Clustering sequential data employing similarity measures on state sequences was used in [4, 8]. These state sequences can be aggregated into Markov Chains modeling the state transitions [17]. HMM have been employed to extract stable groups from temporal data by joint optimization of the model parameters and the cluster count [18].

While the previous work discussed above analyze student clusters at a given point in time, a temporal analysis would allow to identify how interaction patterns change over time and how groups of similar students evolve. Temporal effects of cluster evolution have been analyzed in [15], based on static clustering at each time step. Static approaches are sensitive to noise in the data and may result in temporally inconsistent clusters. Evolutionary clustering methods [7] address this problem as they consider multiple subsequent time steps. The temporal smoothing increases the resulting cluster stability notably and allows for a better analysis of the clusters, i.e., the student properties and interaction patterns. Recently, an evolutionary clustering approach called AFFECT [27] has been introduced that smooths proximities of students over time followed by static clustering. AFFECT was shown to outperform static clustering algorithms.

In this paper, we present a complete processing pipeline for evolutionary clustering that can be used as a black box for any ITS. We incorporate a variation of the AFFECT method into our pipeline and demonstrate that temporal smoothing has beneficial properties for extracting student behavior and groups from educational data. We propose several extensions of the original method tailored towards learning data. Our approach is articulated in four steps. In a first step, we extract action sequences from ITS log data and aggregate them using Markov Chains. We show that the Markov Chain representation of the actions is superior to direct sequence mining techniques [4, 17] with respect to noise cancellation and the ability to identify groups of students with similar behavior. The second step consists of computing pairwise similarities between the Markov Chains. While the proposed pipeline provides flexibility in the choice of similarity measure, the Hellinger distance outperforms other metrics that

are frequently used in the educational data mining literature [4, 17]. Based on the obtained similarities, evolutionary clustering [27] is performed in the third step. The temporal aspect of the student data leads to changing behavior patterns, i.e., we expect the number of clusters and cluster sizes to change over time. Therefore, capturing cluster evolution events, such as merging, splitting, dissolving and forming of clusters, is crucial in order to analyze sequential data. To capture these events automatically, we compute the optimal cluster count for each time step using the AICc criterion.

Using synthetic data, we demonstrate that our method exhibits a higher performance and is more robust to noise than previous work [4, 17]. We further show that our pipeline is able to extract stable clusters over time and reliably detects all cluster events. In an exploratory analysis on real-world data, we apply our pipeline to log data from two different ITS: One for spelling learning and one for mathematics learning. Finally, we present a set of visual tools that are powerful to analyze temporal data and student clusters.

2. METHOD

Our method for student clustering is designed to address two challenges when clustering temporal data. First, the method provides temporally consistent clusters. Second, our pipeline is able to capture changes in cluster sizes as well as in the number of clusters. Four *cluster events* are of particular interest in the context of educational data mining: merging, splitting, dissolving and forming of clusters. If the behavior of students from two different clusters becomes more similar over time, we expect the clusters to *merge* (this could mark a training effect). If on the other hand the behavior of students in a cluster sufficiently diverges clusters might *split* (this could mark the development of different learning strategies). If a distinct behavior disappears within a group of students, we assume the cluster will *dissolve*, meaning students will uniformly change to other clusters. In contrast, *forming* clusters have the potential to mark the development of distinct strategies within students.

The resulting clustering pipeline addressing these challenges is illustrated in Figure 1. The only input required are action sequences extracted from student log data. These action sequences are transformed into Markov Chains for every session and pairwise similarities between these chains are computed. Students are clustered based on these similarities while enforcing temporal consistency over consequent training sessions. Finally, we compute the optimal number of clusters for each training session.

Action Sequences. In a first step we extract action sequences $A_u^t = (a_0, a_1, \ldots, a_n)$ for every session t of a user u. To do so, we map events in the log files of an ITS (e.g. correct/incorrect inputs or help calls) to the actions a_i. As the particular actions depend on the ITS, the extraction of actions has to be changed depending on the ITS.

Action Processing. While action sequences provide rich temporal information about the exact ordering of actions, we expect that they exhibit a considerable amount of noise. We therefore transform the action sequences into an aggregated representation using Markov Chain models, similar to [17]. Markov Chains provide an aggregated view of the pairwise transition probabilities of actions and can be fully described by these transition probabilities $t_{i,j} := p_{a_j|a_i}$ from any state a_i (in our case an action) to any other state a_j. Markov Chains can be extracted using maximum likelihood estimates of the transition probabilities $t_{i,j}$.

Similarity Computation. To cluster student behavior, a suitable similarity (or distance) measure between students has to be defined. In educational data mining, popular choices for measuring distances between action sequences are the longest common subsequence (LCS) and the Levenshtein distance (see e.g. [4]). LCS measures the length of the largest set of characters that appear in left-to-right order within the string, not necessarily at consecutive places. The Levenshtein distance computes the number of insertions, deletions and replacements needed to transform one string into the other. Instead of computing distances directly on action sequences we can apply the computation to the aggregated values of Markov Chains. Previous work [17] has been using the Euclidean distance between the transition probabilities of two Markov Chains. A potential disadvantage of the Euclidean distance is that it is not designed for the comparison of probabilities. Therefore, we propose to use metrics that are specifically designed for comparing probability distributions. Since the conditional probabilities describing a Markov Chain do not form a proper probability distribution (the entries of the transition probability matrix do not sum up to one), we compute the expected transition probabilities using the stationary distribution over the actions and compare these expected transition frequencies $\bar{t}_{i,j}$ instead of the conditional probabilities $t_{i,j}$. We use two common metrics: the Jensen-Shannon Divergence and the Hellinger distance [21] to compute the distances between the expected transition frequencies $\bar{t}_{i,j}$ of the Markov Chains.

Clustering. Using the measures defined above we compute a pairwise similarity matrix W^t for every session t of the training (entries of the matrix measure how similar two students are during that particular training session). These similarity matrices can then be clustered by any standard clustering method. However, clustering students for each session individually does not make use of the temporal information available. Recently, a method for clustering evolutionary data has been proposed that accurately tracks the time-varying similarities of objects over discrete time steps [27]. The method assumes that the observed similarities W^t are a linear combination of the true similarity between students Ψ^t and random noise N^t:

$$W^t = \Psi^t + N^t. \tag{1}$$

Instead of performing clustering directly on W^t, a smoothed similarity matrix $\hat{\Psi}^t$ is proposed, given as

$$\hat{\Psi}^t = \alpha^t \hat{\Psi}^{t-1} + (1 - \alpha^t) W^t, \tag{2}$$

where α^t controls the amount of smoothing applied to the observed similarity matrix W^t. Under some assumptions (detailed in [27]) an optimal choice for α^t is

$$\alpha^t = \frac{\sum_i \sum_j \mathrm{var}(n_{ij}^t)}{\sum_i \sum_j (\hat{\psi}_{ij}^{t-1} - \psi_{ij}^t)^2 + \mathrm{var}(n_{ij}^t)}. \tag{3}$$

This means that the optimal α^t is based on a trade-off between the estimated noise in W^t and the amount of new information that W^t contains compared to previous similarity matrices. If W^t exhibits a lot of noise we more heavily rely on previous observations (high α^t) but if we observe large

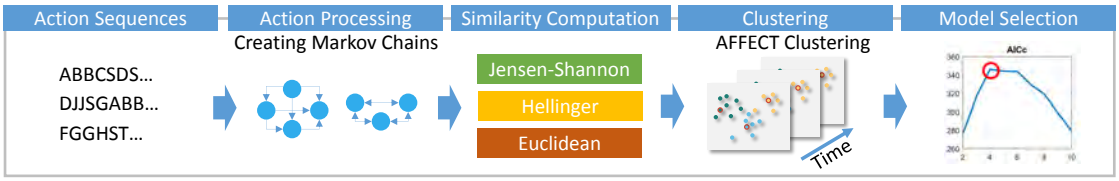


Figure 1: Overview of our clustering pipeline. Action sequences are extracted from log data and transformed into Markov Chains per session. Pairwise similarities between students are computed for every session. Clustering is performed using evolutionary clustering [27]. Finally, the AICc criterion selects the best model.

discrepancies between the previous similarity estimates and the current ones (e.g. some students show a novel behavior) we emphasize the similarities from the current session (low α^t). Finally, we use the standard clustering algorithm K-Means to cluster the smoothed similarity matrices $\hat{\Psi}^t$.

Model Selection. The assumption of temporal consistency in the pairwise similarities between students does not prohibit evolution of clusters if students change their behavior over the course of the training. Such long-term drifts lead to growing and shrinking of clusters eventually, and even to dissolving and forming of clusters over time. In contrast to the original AFFECT method [27], we therefore compute the optimal number of clusters in every time step. Deciding on the number of clusters is a variant of the model selection problem, for which various different criteria exists. The Akaike information criterion (AIC) and the Bayesian information criterion (BIC) are among the most common criteria for model selection. The main difference between BIC and AIC is that the BIC penalizes the number of clusters more strongly than AIC. AICc corrects the AIC criteria for finite sample sizes. For our experiments, we used AICc as it potentially reveals more clusters, which is important for our exploratory analysis of learning data. To compute the AICc the log likelihood (LL) of the model is needed. According to [23], the LL for K-Means can be formulated as

$$LL = \sum_i \log(\frac{N_{c(i)}}{N}\phi(x_i|\mu_{c(i)},\sigma)), \quad (4)$$

where N denotes the number of samples, $c(i)$ the cluster index of sample x_i and $N_{c(i)}$ the number of samples in cluster $c(i)$. The likelihood of a sample x_i that was assigned to cluster $c(i)$ can be computed using the probability distribution $\phi(x_i|\mu_{c(i)},\sigma)$, where $\mu_{c(i)}$ denotes the centroid of the cluster and σ the empirical variance of the data. In our case (as suggested by [23]), the probability distributions ϕ are identical spherical Gaussians. To compute the LL, we embed our data points in a Euclidean space in which the distances between the points match the similarities extracted from the action sequences. To perform this embedding, we use the method presented in [12] that transforms N objects with pairwise similarities to a $D = N - 1$ dimensional Euclidean space.We then estimate the effective dimensionality $\hat{D}$ of our data set as the sum of eigenvalues λ_i of the covariance matrix divided by the largest eigenvalue λ_1 (see [16]): $\hat{D} = \sum_i \lambda_i/\lambda_1$. This means that the effective number of parameters P for the K-Means clustering is $P = (\hat{D} + 1)k$, where k is equal to the number of clusters (see e.g. [23] for a derivation). Based on the LL and the estimated effective dimensionality of our data $\hat{D}$, we calculate the AICc as $-2LL + 2P + (2P(P + 1))/(n - P - 1)$.

3. SYNTHETIC EXPERIMENTS

We analyzed the properties of our clustering algorithm using synthetic data and we compared the performance and stability of our method to previous algorithms for clustering sequential educational data. Finally, we also validated our model selection step.

Experimental setup. We simulated student actions for 80 students over 50 sessions in a simulated learning environment. Students needed to solve 20 tasks per session. Student abilities θ and task difficulties d were simulated as part of a Rasch model [26]. Student abilities for all students were sampled from a normal distribution with mean μ and variance σ. Task difficulties were sampled uniformly from the range $[-3, 3]$ in agreement with the common range of task difficulties [10]. Each task y consisted of eight steps s_j that a student had to complete to finish the task (this could e.g. be letters of a word to spell, performing steps of a calculation or solving a physics problem). The probability of a student correctly solving a task was then given by the Rasch model as $p(y) = (1 + e^{-(\theta-d)})^{-1}$. In our simulation (in accordance with many ITS) a task was correctly solved if all the substeps are correctly solved, which defines the probability of correctly solving a step of a task s_j to be $p(s_j) = (p(y))^{\frac{1}{8}}$. Finally, a student could request help at any point in time during the training. Whether the student asked for help was sampled from a Bernoulli distribution with p_H. Based on the described sampling procedure we emitted the following actions for a student: *new task*, *help*, *correct*, *incorrect*, *correction*, *task completed*. The number of sampled actions per student and session depended on the performance of the student (e.g. a student who gets every step of a task correct completes a task after eight *correct* actions, whereas another student who requests help and commits an error requires more actions to complete the task).

For our experiments we simulated student groups with different behavior. For the chosen range of task difficulties, student abilities are found to be normally distributed with mean $\mu = 0$ and variance $\sigma = 1$ (see [10] for details). We simulated good performing students by setting $\theta = 1$ and bad performing students by setting $\theta = -1$. According to [2], the most frequent form of help abuse are multiple consecutive help requests. We simulated this behavior by a large probability $p_H = 0.2$ to ask for help instead of working on the task, while normal help seeking behavior has a smaller probability for requesting help $p_H = 0.05$. Based on these different properties we simulated four groups of 20 students as follows. Group A contains bad performing students ($\theta = -1$) that rarely ask for help ($p_H = 0.05$). Group B consists of bad performing students ($\theta = -1$) that frequently use the help system ($p_H = 0.2$). Group C and D consist of

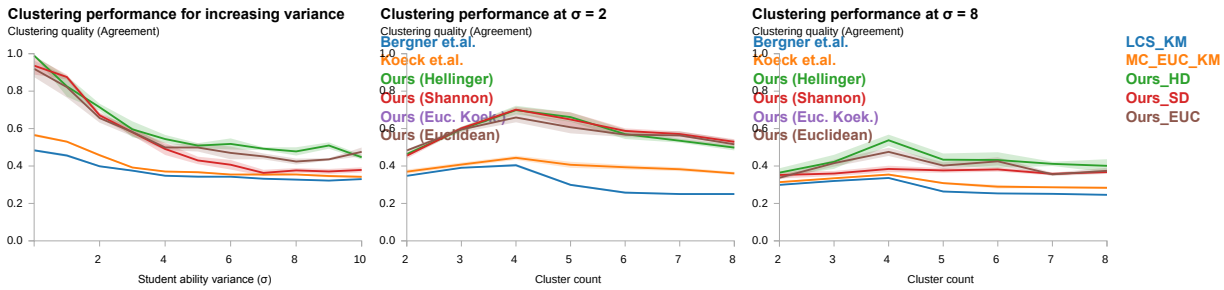


Figure 2: Comparison of clustering methods over increasing noise levels (left) and over different numbers of clusters for fixed noise levels $\sigma = 2$ (middle) and $\sigma = 8$ (right). Our method (*Ours_HD*, *Ours_SD*, *Ours_EUC*) shows less degradation of clustering quality (agreement with ground truth) for increasing noise levels.

good performing students ($\theta = 1$) with rare ($p_H = 0.05$) and frequent ($p_H = 0.2$) help requests, respectively.

Our proposed pipeline offers flexibility in the choice of the similarity measure (see Section 2). We used the Jensen Shannon divergence [21], the Hellinger distance [21] and the Euclidean distance for our experiments, and refer to these approaches as *Ours_SD*, *Ours_HD*, and *Ours_EUC*. To measure the influence of the different elements of the pipeline on the overall performance, we compared the proposed method to previous work on clustering of action sequences. The first approach [4] works directly on the action sequences and uses the longest common subsequences (LCS) as similarity measure. Clustering is performed using an agglomerative clustering. However, to be able to better compare clustering results we used the proposed similarity measure together with K-Means. We refer to this pipeline as *LCS_KM*. Similar to our method, the second approach used for comparison [17] computes the similarities between students using Markov Chains. Similarities are measured using the Euclidean distance and clustering is performed using K-Means. The pipeline for this approach is denoted by *MC_EUC_KM*.

Clustering Quality & Robustness. In a first experiment, we computed the clustering quality of the different approaches with increasing noise levels. The performance P was measured using the cluster agreement in comparison to the ground truth labels. The different noise levels were simulated by increasing the variance in student abilities σ for the sampling of the data. Figure 2 (left) illustrates the performance of the different approaches with increasing noise. Note that the performance was computed using the correct cluster count of $k = 4$. Our pipeline (colored in green, red, and brown) exhibits the highest performance over all noise levels. The average agreement of our best performing pipeline (P_{Ours_HD}) is substantially higher than the average agreement of the best previous approach ($P_{MC\_EUC\_KM}$), both for a low variance ($P_{Ours\_HD,\sigma=1} = 0.82$, $P_{MC\_EUC\_KM,\sigma=1} = 0.53$) and for noisy data ($P_{Ours\_HD,\sigma=10} = 0.45$, $P_{MC\_EUC\_KM,\sigma=10} = 0.34$).

To investigate these differences between the approaches, we measured their performance over different numbers of clusters at preset noise levels. Figure 2 (middle) illustrates the results for data with a relatively low noise level ($\sigma = 2$), while Figure 2 (right) shows the clustering quality of the different pipelines on noisy data ($\sigma = 8$). In the case of small noise in the data, all methods exhibit the best performance for the correct number of clusters ($k = 4$), which is a desirable property. The results demonstrate that using Markov Chains ($P_{MC\_EUC\_KM,k=4} = 0.44$) instead of working directly on action sequences ($P_{LCS\_KM,k=4} = 0.40$) leads to a higher clustering quality. A further increase in performance is achieved by our proposed algorithm: The variations of our pipeline exhibit a substantially higher clustering quality ($P_{Ours\_EUC,k=4} = 0.66$, $P_{Ours\_HD,k=4} = 0.70$, $P_{Ours\_SD,k=4} = 0.70$) than the previous work. This substantial increase in performance ($\Delta P_{k=4} = 0.26$ compared to MC_EUC_KM) is due to two changes in the pipeline. First, the proposed pipeline uses the AFFECT method for clustering leading to an increase in performance of $\Delta P_{k=4} = 0.20$. Second, while MC_EUC_KM computes the similarity measure directly on the transition probabilities, we use the expected transition probabilities as a basis for the similarity computations (see Section 2) accounting for an improvement in performance of $\Delta P_{k=4} = 0.06$. Within our approach, the choice of similarity measure has only a small impact on the clustering quality. Figure 2 (right) demonstrates that our proposed method is more robust to noise than previous work [17, 4]. The best variation of our pipeline (colored in green) still achieves a reasonable performance ($P_{Ours\_HD,\sigma=8} = 0.54$). At these noise levels, the choice of action processing (Markov Chains vs. direct processing of action sequences) does not significantly influence performance ($P_{LCS\_KM,k=4} = 0.34$, $P_{MC\_EUC\_KM,k=4} = 0.35$). The choice of the clustering algorithm on the other hand is important. The increased performance of our method can be attributed to the use of AFFECT for clustering: AFFECT takes into account data from previous time steps to perform the clustering. Interestingly, the pipeline using the Jensen Shannon divergence (*Ours_SD*) seems less robust to noise than the other pipelines (*Ours_HD* and *Ours_EUC*).

Stability. When clustering student actions over time, temporal consistency of clusters is essential. We measured the temporal stability of our method by computing the cluster size over the 50 simulated sessions (see Figure 3). We compared the best performing pipeline from the first experiment (*Ours_HD*) to the previous approaches (*LCS_KM*, *MC_EUC_KM*) using again $k = 4$ clusters. As can be seen from Figure 3 (left), our method provides a smooth temporal clustering with stable cluster sizes over time. The clusters found by *MC_EUC_KM* (Figure 3 (middle)) and *LCS_KM* (Figure 3 (right)), on the other hand, are unstable: cluster sizes vary significantly over time. These results are as expected, as static clustering approaches identifying groups of students at each point in time are very sensitive to

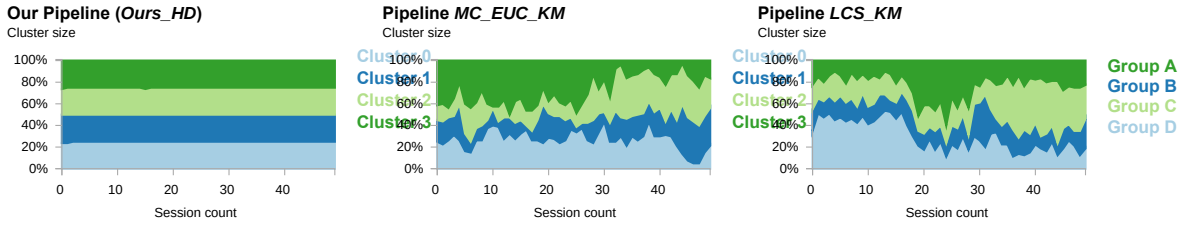


Figure 3: Relative cluster sizes (for $k = 4$ clusters) over 50 simulated sessions. Our method performs best in extracting temporally stable clusters.

noise. The proposed method solves this problem by applying an evolutionary clustering algorithm and therefore takes into account multiple time steps.

Interpretability. Since we are clustering student behavior over multiple sessions, we expect the number of clusters and the cluster sizes to change over time. We expect clusters to merge, split, dissolve and form (see Section 2 for details). We evaluated the *Ours_HD* pipeline on four scenarios using synthetic data. Note that these scenarios are artificial and are used only to demonstrate that the pipeline can capture the described events; we will show real-world examples of these events in Section 4. In the first scenario (Figure 4 (top left)), group A consisting of bad performing students with rare help calls (colored in dark green) merges into group B (colored in dark blue), i.e. the students of group A also start abusing the help. In our simulation, we start the cluster merge after $t = 20$ sessions and let group A completely vanish after $t = 50$ sessions, a behavior that is nicely captured by our method. The second scenario (Figure 4 (top right)) starts with only three groups (B, C, and D), assuming that all bad performing students frequently use the help. Over time, the bad performing students split into a group abusing the help (group B, colored in dark blue) and a cluster consisting of students with rare help calls (group A, colored in dark green), i.e. in the simulation some of the bad performing students stop abusing the help over time. In the third scenario (Figure 4 (bottom left)) a dissolving cluster is simulated: Over time, group B (colored in dark blue) completely dissolves and the students are distributed to the other three clusters. The fourth scenario (Figure 4 (bottom right)), finally, simulates a forming cluster event. The simulation starts with only three clusters (groups A, C, and D). With an increasing number of sessions, a fourth cluster forms (group B, colored in dark blue) and students from the other three clusters slowly switch to the new cluster until all the groups have equal size (after $t = 50$ sessions). This event is again correctly captured by our method. The presented experiments demonstrate that the proposed pipeline is able to reliably identify changing cluster numbers and sizes. The results also demonstrate the validity of the model selection step of the pipeline: The AICc correctly identifies the number of clusters for all scenarios.

4. EXPLORATORY DATA ANALYSIS

We applied our method to clustering of student interactions from two different ITS, focusing on the identification and interpretation of *cluster events*.

Experimental Setup. The first data set contains log data from 106 students and was collected using *Orthograph*, a

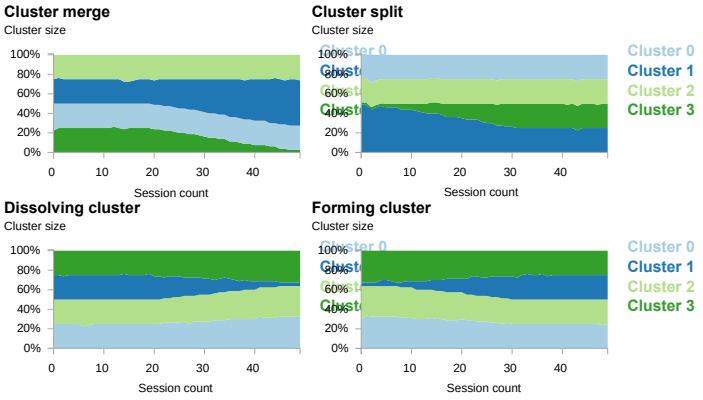


Figure 4: Simulated examples of four types of *cluster events*. Our pipeline correctly identifies cluster merges/splits as well as dissolving/forming clusters.

computer-based training program for elementary school children with dyslexia [9]. *Orthograph* consists of one main learning game, where children have to type a dictated word. The second data set contains data from 134 students and was collected from *Calcularis*, an ITS for elementary school children with difficulties in learning mathematics [13]. *Calcularis* consists of different games for training number representations and calculation. For all students, we extracted the first 15 training sessions with a minimal duration of $t = 5$ minutes from each student.
All results have been computed using our pipeline *Ours_HD* (see Section 2), applying the Hellinger Distance to measure similarities between Markov Chains of different students.

Navigation Behavior. In a first experiment, we extracted actions describing the *Navigation Behavior* of children in *Orthograph*. *Navigation Behavior* captures all events that cause the displayed content to change. During game play, children collect points for correct responses as well as for time spent in the training in general. These points can be used to buy different visual perks for the game in the shop. Children can also analyze their performance (e.g. progress in the current module) in the progress view. The resulting Markov chain (see Figure 5) consists of three possible states: *Game*, *Shop*, and *Performance*.

Figure 6 shows the relative cluster sizes for the *Navigation Behavior* Markov Chain over the first 15 sessions of the training. The different colors denote different clusters. At the beginning of the training ($t = 0$), our pipeline detects seven different clusters, however, three of these clusters (col-

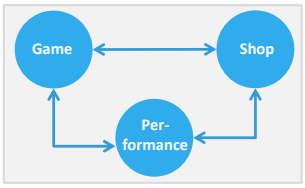


Figure 5: Markov Chain for actions that capture the *Navigation Behavior* of students in *Orthograph*.

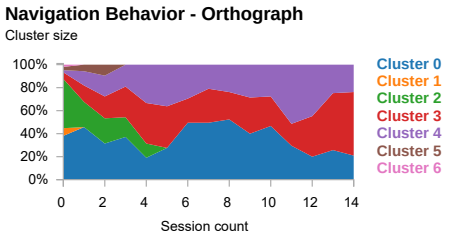


Figure 6: Relative cluster sizes based on the *Navigation Behavior* extracted from *Orthograph*.

ored in pink, brown, and orange) die within the first three training sessions. Children in these clusters spent more than 50% of their time browsing the shop and checking their performance (orange: 46% *Game*, 31% *Shop*, 23% *Performance*; brown: 43% *Game*, 22% *Shop*, 35% *Performance*; pink: 40% *Game*, 32% *Shop*, 28% *Performance*) at the beginning of the training. We therefore hypothesize that children in these clusters tried out and played with the different views before getting used to the navigation possibilities of the system.

After $t = 5$ time steps, a further cluster (colored in green) dissolves before the clustering stabilizes to three main groups (colored in blue, red, and purple). Figure 7 (top) shows the transition probabilities of the Markov Chains for the different clusters before the clusters dissolve (after $t = 3$ sessions). Children in the blue cluster are very focused on training, they spend 82% of their time in the *Game*. Once in the *Shop* or *Performance* state (18% of their time) they tend to select the following view with equal probabilities. Children in the red cluster like to browse the shop, a behavior that is visible from the high transition probabilities to the *Shop* state (*Game*→*Shop*: 0.41; *Performance*→*Shop*: 0.39), resulting in 34% of the training time spent browsing the shop. The purple cluster consists of children, who like to navigate to the shop and performance overview between solving the different tasks (*Game*→*Shop*: 0.41, *Game*→*Performance*: 0.44). However, these tend to be shorts visit as they will return to playing the game right after with high probability (*Performance*→*Game*: 0.58, *Shop*→*Game*: 0.77). Finally, children in the green cluster tend to select the next view randomly when playing the game. Once in the *Performance* state, they have a probability of 0.30 to browse the shop right after. The analysis of this time step illustrates that the different clusters differentiate well between focused children not making use of the navigation possibilities (blue cluster), children who frequently (but reasonably) use the different views (purple and green cluster), and distracted children who spend long amounts of time off-task (red cluster).

After $t = 6$ training sessions, the green cluster dissolves and students from this cluster change to the red and blue clusters. The transition probabilities of the Markov Chains for these stable main clusters are illustrated in Figure 7 (bottom). The children in the blue cluster are still focused on training, spending 76% of their time solving tasks. However, they also check their training progress from time to time (14% of the time spent in the *Performance* state). After checking training progress, they tend to also browse the shop (*Performance*→*Shop*: 0.27). The children in the purple cluster have stopped navigating to the performance overview between different tasks (*Game*→*Performance*: 0.17) and instead visit the shop more frequently (*Game*→*Performance*: 0.58) and longer (35% of time spent in the *Shop* state). The red cluster still consists of children who like browsing the shop, a behavior that is visible from the high transition probabilities to the *Shop* state (*Game*→*Shop*: 0.33; *Performance*→*Shop*: 0.31). However, they also tend to spend time checking their progress, resulting in 47% of the training time spent off-task. Students from the green cluster therefore changed their behavior from frequent, but short off-task navigation to a more focused training style (change to blue cluster) or to being completely distracted and spend long amount of times off-task (change to the red cluster).

Input & Help Seeking Behavior. Our method can be used as a black box for any ITS and therefore also allows for comparison of behavior patterns across different ITS. The only user input needed is the definition of possible actions. To illustrate this possibility, we extracted two different sets of actions *Input Behavior* and *Help Seeking Behavior* from data collected with *Orthograph* and with *Calcularis*.

Input Behavior captures all possible inputs. Implicitly these actions capture the performance of students, as e.g. a bad performing student is likely to commit more mistakes. In *Orthograph*, children train spelling by writing words that are played back by the system. Therefore, the *Input Behavior* Markov Chain for *Orthograph* (see Figure 8) consists of four states: Children can type a letter (*Input*), correct themselves by deleting a letter (*Backspace*), provide invalid input such as typing a number (*Invalid Input*), or submit their solution (*Enter*). For *Calcularis*, we investigated calculation games. In these games, children need to solve different mental addition and subtraction tasks. We again define four states for the *Input Behavior* Markov Chain (see Figure 8): children can type a digit (*Input*), correct themselves by deleting a digit (*Correction*), provide invalid input such as random mouse clicks (*Invalid Input*), or set their answer (*Enter*).

Figure 9 shows the relative cluster sizes for the *Input Behavior* action set from *Orthograph* over 15 training sessions. Our method identifies three stable clusters. Investigating the stationary distributions of the Markov Chains reveals that students in the orange cluster show the highest probabilities for committing invalid inputs over all sessions ($t=3$: 0.15; $t=7$: 0.23; $t=13$: 0.16). The green cluster consists of focused students who consistently produce a low percentage of invalid inputs ($t=3$: 0.06; $t=7$: 0.04; $t=13$: 0.05). Students in the blue cluster also tend to show low probabilities for invalid inputs across the different sessions ($t=3$: 0.11; $t=7$: 0.09; $t=13$: 0.08). The orange cluster is an example of a *forming cluster* growing in size over the course of the training. We hypothesize that this event marks the increasing difficulty of the tasks and is caused by a downwards drift

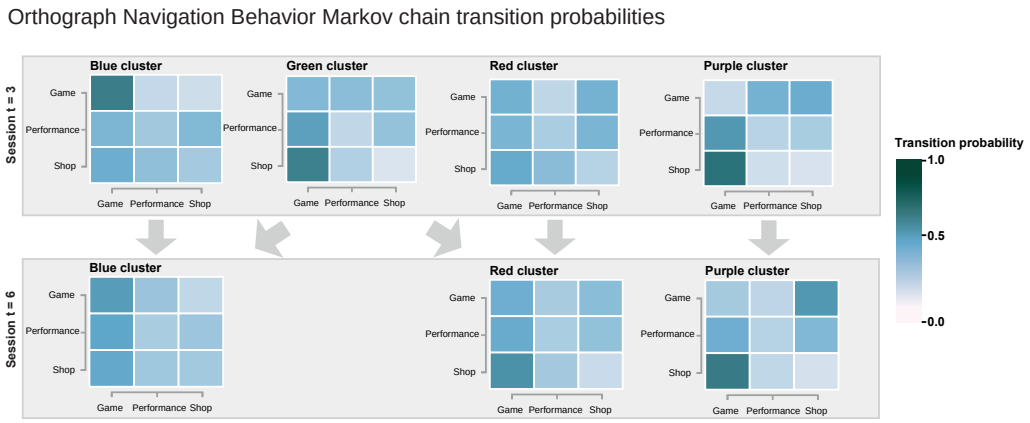


Figure 7: Transition probabilities of the average Markov Chain for each cluster present in session $t = 3$ (top) and session $t = 6$ (bottom) for *Navigation Behavior* in *Orthograph*. The arrows indicate students transferring from the green cluster to the blue and red clusters between session $t = 3$ and session $t = 6$.

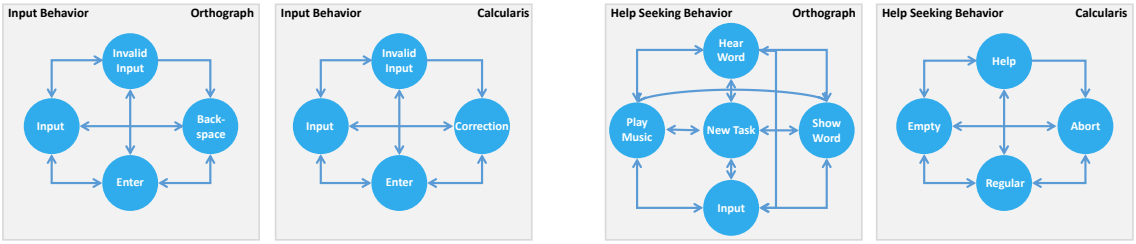


Figure 8: Markov Chains for the *Input Behavior* and the *Help Seeking Behavior* in *Orthograph* and *Calcularis*.

of students from the clusters with good performing students to the clusters with students showing worse performance. Further analysis of cluster transfers reveals that students indeed are never switching directly from the green (best performance) to the orange cluster (worst performance).

For *Calcularis*, the *Input Behavior* clusters are relatively stable over the course of the training (see Figure 9). There is one distinct *dissolve event* in the first four sessions: the orange cluster is dissolving into the blue and green clusters. Investigating the stationary distributions of the Markov chains of the three clusters reveals that all clusters have a relatively low probability for invalid inputs (t=2: 0.17 (blue), 0.12 (orange), 0.08 (green)). However, students belonging to the blue cluster tend to perform multiple consecutive corrective actions in a row (*Correction*→*Correction*: 0.25 (blue), 0.13 (orange), 0.13 (green)). Students in the orange cluster are most likely to enter a valid input after a correction (*Correction*→*Input*: 0.68 (orange), 0.57 (blue), 0.65 (green)).

In *Orthograph*, differences in *Input Behavior* are mainly expressed by the percentage of invalid inputs provided. We observe a more distinct picture for *Calcularis*. While the invalid inputs are still an important indicator, children also exhibit different corrective behaviors.

Help Seeking Behavior captures the use of hints available in the training environment. In *Orthograph*, children can re-play the given word (*Hear Word*), play the melody of the word (*Play Music*) and show the correct spelling of the word (*Show Word*). The according Markov Chain is displayed in Figure 8. The states *New Task* and *Input* denote the play-back of a new word and a user input (keyboard), respectively. The development of the relative cluster sizes for these action sequences (see Figure 9) reveals a surprisingly large variance in student behavior (the clustering algorithm finds nine different clusters in the first two training sessions). However, the diversity in student behavior disappears through a large *cluster merge* after $t=3$ sessions. Investigating the transition probabilities between the different actions, we observe that while students are experimenting with the three different help systems at the beginning of the training, the final cluster of students gave up on using the help functions. This drop in the frequency and diversity of help usage indicates that the help functionality provided in *Orthograph* is not useful for most of the students.

Calcularis provides a limited help functionality. Children can require explanations for games (*Help*). Furthermore, they can directly require the solution of a task (*Empty*), if the task seems too difficult. Further states of the Markov Chain (displayed in Figure 8) are the setting of a complete answer (*Regular*) and the abortion of a task (*Incomplete*). We again observe a large *cluster merge* at the beginning of the training leading into two stable clusters. Investigating the stationary distributions of the Markov Chains of the two clusters reveals that students in the orange cluster are more likely to perform a help request compared to the blue cluster (t=6: 0.03 (blue), 0.13 (orange)).

The *Help Seeking Behavior* of the children is more difficult to compare across different ITS, because the available hints are very different. However, our experiment shows that both learning environments do not provide ideal help options.

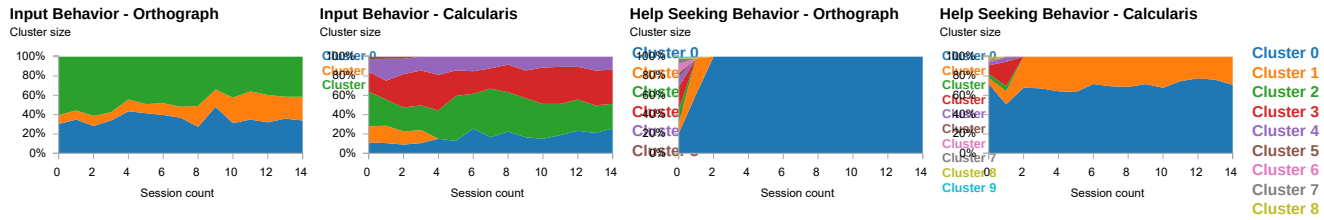


Figure 9: Relative cluster sizes over the first 15 sessions based on the clustering of *Input Behavior* (left) and *Help Seeking Behavior* (right) for students training with *Orthograph* and *Calcularis*.

5. CONCLUSIONS

We presented a complete pipeline for the evolutionary clustering of student behavior. This pipeline can be used as a black box for any ITS, requiring only the extraction of action sequences as input. We demonstrated that enforcing temporal coherency between consecutive clusterings is beneficial for the detection of student behavior as well as the stable detection of *cluster events*. Our method outperforms previous work on synthetic data regarding clustering quality and stability. We applied our pipeline to different types of action sequences collected from two different ITS. The exploratory analysis demonstrates that our method is able to reveal interesting properties about the behavior of students and potential deficiencies of the learning environments.

Acknowledgments. This work was supported by ETH Research Grant ETH-23 13-2.

6. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Data Engineering*. IEEE, 1995.

[2] V. Aleven, B. McLaren, I. Roll, and K. Koedinger. Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *IJAIED*, 2006.

[3] J. M. L. Andres, M. M. T. Rodrigo, R. S. Baker, L. Paquette, V. J. Shute, and M. Ventura. Analyzing Student Action Sequences and Affect While Playing Physics Playground. In *AMADL*, 2015.

[4] Y. Bergner, Z. Shu, and A. A. Von Davier. Visualization and Confirmatory Clustering of Sequence Data from a Simulation-Based Assessment Task. In *Proc. EDM*, 2014.

[5] G. Biswas, H. Jeong, J. S. Kinnebrew, B. Sulcer, and R. ROSCOE. Measuring self-regulated learning skills through social interactions in a teachable agent environment. *RPTEL*, 2010.

[6] K. E. Boyer, R. Phillips, A. Ingram, E. Y. Ha, M. Wallis, M. Vouk, and J. Lester. Characterizing the Effectiveness of Tutorial Dialogue with Hidden Markov Models. In *Proc. ITS*, 2000.

[7] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proc. KDD*, 2006.

[8] M. Desmarais and F. Lemieux. Clustering and visualizing study state sequences. In *Proc. EDM*, 2013.

[9] M. Gross and C. Vögeli. A multimedia framework for effective language training. *Comput. & Graph.*, 2007.

[10] D. Harris. Comparison of 1-, 2-, and 3-parameter IRT models. *Educational Measurement*, 1989.

[11] J. Herold, A. Zundel, and T. F. Stahovich. Mining meaningful patterns from students' handwritten coursework. In *Proc. EDM*, 2013.

[12] T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *Pattern Analysis and Machine Intelligence*, 1997.

[13] T. Käser, G.-M. Baschera, J. Kohn, K. Kucian, V. Richtmann, U. Grond, M. Gross, and M. von Aster. Design and evaluation of the computer-based training program calcularis for enhancing numerical cognition. *Frontiers in Developmental Psychology*, 4(489), 2013.

[14] J. S. Kinnebrew and G. Biswas. Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution. In *Proc. EDM*, 2012.

[15] J. S. Kinnebrew, D. L. Mack, and G. Biswas. Mining temporally-interesting learning behavior patterns. In *Proc. EDM*, 2013.

[16] M. Kirkpatrick. Patterns of quantitative genetic variation in multiple dimensions. *Genetica*, 2006.

[17] M. Köck and A. Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *UMUAI*, 2011.

[18] C. Li and G. Biswas. A Bayesian Approach to Temporal Data Clustering using Hidden Markov Models. In *ICML*, 2000.

[19] R. Martinez-Maldonado, K. Yacef, and J. Kay. Data mining in the classroom: Discovering groups' strategies at a multi-tabletop environment. In *Proc. EDM*, 2013.

[20] L. D. Miller and L.-K. Soh. Meta-Reasoning Algorithm for Improving Analysis of Student Interactions with Learning Objects using Supervised Learning. In *Proc. EDM*, 2013.

[21] L. Pardo. *Statistical inference based on divergence measures*. CRC Press, 2005.

[22] T. Peckham and G. McCalla. Mining Student Behavior Patterns in Reading Comprehension Tasks. In *Proc. EDM*, 2012.

[23] D. Pelleg and A. Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *Proc. ICML*, 2000.

[24] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaïane. Clustering and sequential pattern mining of online collaborative learning data. *TKDE*, 2009.

[25] Y. Wang and N. T. Heffernan. The student skill model. In *Proc. ITS*, 2012.

[26] M. Wilson and P. De Boeck. Descriptive and explanatory item response models. In *Explanatory item response models: A generalized linear and nonlinear approach*. Springer, 2004.

[27] K. S. Xu, M. Kliger, and A. O. Hero Iii. Adaptive evolutionary clustering. *Data Mining and Knowledge Discovery*, 2014.

Web as a textbook: Curating Targeted Learning Paths through the Heterogeneous Learning Resources on the Web.

Igor Labutov
Cornell University
iil4@cornell.edu

Hod Lipson
Columbia University
hod.lipson@columbia.edu

ABSTRACT

A growing subset of the web today is aimed at *teaching* and *explaining* technical concepts with varying degrees of detail and to a broad range of target audiences. Content such as tutorials, blog articles and lecture notes is becoming more prevalent in many technical disciplines and provides up-to-date technical coverage with widely different levels of prerequisite assumptions on the part of the reader. We propose a task of organizing heterogeneous educational resources on the web into a structure akin to a textbook or a course, allowing the learner to navigate a sequence of webpages that take them from point A (their prior knowledge) to point B (material they want to learn). We approach this task by 1) performing a shallow term-level classification of what concepts are *explained* and *assumed* in any given text, and 2) using this representation to connect web resources that explain concepts to those web resources where the same concepts are assumed. The main contributions of this paper are 1) a supervised classification approach to identifying explained and assumed terms in a document and 2) an algorithm for finding optimal paths through the web resources given the constraints of the user's goal and prior knowledge.

Keywords

web resources; optimizing learning

1. INTRODUCTION

No scholar is born at the frontier of knowledge — early learning and lifelong learning both play a defining role in shaping the research vector of an academic [7]. More alarming, recent research [6] demonstrates that the pre-career idle time of an up-and-coming researcher has been on the steady rise during the last century, attributing to the "burden of knowledge" phenomenon — the inflation of the body of prerequisite prior knowledge to be mastered before being able to contribute to the field with original research. The hypothesis of [9, 10] is that facilitating effective early and lifelong learning practices is a viable way for easing the "burden of knowledge".

While physical textbooks and classrooms traditionally assumed the role of knowledge curators, they also present a bottleneck in today's rapidly growing web of up-to-date technical and academic content — peer-reviewed articles, lecture notes, tutorials, slides etc — from academics and "citizen scientists" alike. An automatic approach for "weaving" natural curricular progressions through the web of such heterogeneous academic/educational content, we believe, will catalyze early and lifelong learning by creating more efficient and goal-oriented curricula targeted to the level of the audience.

The web is the only collection of resources today where attempting this task becomes meaningful and promising. The reason for this is that the web contains an extensive amount of diversity in its content, i.e. content that explains the same concepts but in many different ways. Naturally this diversity reflects the diversity of the people who create this content, their backgrounds, styles of learning and ways of thinking about complex concepts, which would naturally match learners with similar characteristics. We believe that this diversity can be leveraged to create learning pathways that are not bound to the traditional curricula that are often constrained for no better than a historical reason. We propose instead to optimize a curriculum directly for *what you want to know* given *what you already know*.

We propose to tackle the problem of *curriculum mining* on the web, which broadly, involves linking technical resources on the web to other resources that explain a subset of concepts that are assumed in the original document. We propose to decompose the task into 1) understanding what is *explained* and *assumed* in a document on the part of the the reader and 2) use this document-level representation to sequence documents that guide the learner from their current state of knowledge towards their goal, for example, understanding a specific research paper or a set of lecture notes.

We propose a *term-centric* approach for inducing curricular relations between any pair of documents. Naturally, understanding a technical concept is more than being familiar with its surface term, and in this view an approach that operates at the level of individual terms may appear to be naïve. After all, to explain a new concept is to put together existing concepts in a novel way [13], and in the process introduce convenient nomenclature. However, we hypothesize, that by the virtue of seeking the shortest sequence of documents that "cover" (explain) multiple terms at once, the resulting bottle-

neck will implicitly "prefer" to link to prerequisite documents that introduce and explain whole concepts, i.e. groups of terms, as opposed to introducing terms one document at a time (an extreme example would be presenting a sequence of pages from a dictionary, each document defining a term independence; this is clearly undesirable). It will be our running assumption, that there exists a correlation between the knowledge of the terms and the understanding of the overarching concept.

Thus, to a first-order approximation, we model technical documents as "bags of terms", and in the interest of tractability set forth the following set of modeling assumptions:

- **Assumption 1** A document is a bag-of-technical-terms (multiset) that is further partitioned into two multisets: E (*Explained*), A (*Assumed*) — corresponding to the role (aspect) of the term within the document:

  Explained: The terms appear in the context that furthers the understanding of the concept corresponding to the term.

  Assumed: The concept corresponding to the term is assumed to be familiar, and is required for understanding the context in which it appears.

- **Assumption 2** The degree of reliance on the knowledge of a particular term in the document is proportional to the frequency of the term in the *Assume* multiset, i.e. which concepts are fundamental to the understanding of the document, and which are auxiliary is reflected in the number of occurences of the corresponding terms.

As an illustration, consider the following excerpt from Christopher Bishop's classic textbook *Machine Learning and Pattern Recognition* from the chapter that introduces the concept of *Expectation Maximization*:

> **Expectation Maximization**
> An elegant and powerful method for finding maximum likelihood solutions for models with latent variables is called the expectation maximization algorithm, or EM algorithm.

In the excerpt above, we solid–underline the terms that appear in the *Explained* aspect and dash–underline terms that appear in the *Assumed* aspect. Understanding the concept of *Maximum likelihood* is a prerequisite for understanding *Expectation Maximization*. It is no surprise that most resources that introduce the concept of *Expectation Maximization* implicitly assume that the reader is familiar with *Maximum Likelihood*. Academic and educational literature is fraught with such implicit assumptions that may be challenging to unravel for a learner especially new to the area. Note that on the surface it may seem that detecting instances of explained terms in the text is an equivalent task to finding instances of term definitions – a well studied task – but it is not so. Especially in technical disciplines, explaining a concept requires much more than giving a definition. A document defining a term, may or may not actually explain the concept behind it. For example, a document may define a term to refresh the reader's memory but otherwise assume the reader's familiarity with it. On the other hand, a document may explain a term without ever giving a one-sentence definition.

Finally, the proposed dichotomy may appear as a gross oversimplification, ignoring the entire continuum of pragmatics between the two extremes. We argue that while binary term-level classification alone may not capture the fine-grained aspect of any one term, combining it with the context of the entire document, will enable us to unravel the prerequisite relationships between documents.

2. RELATED WORK

Evidence of information overload in traditional textbooks Formal study of textbook organization conducted by [1] on a corpus of textbooks from India quantitatively addresses the issue known as the "mentioning problem" [12], where "concepts are encountered before they have been adequately explained and forces students to randomly 'knock around' the textbook". The work of [1] suggests that many traditional textbooks suffer from the resulting phenomenon of "information burden" and provide diagnostic metrics for evaluating it. A user study conducted by [2], though limited to electronic textbooks, demonstrated the utility of a navigational aid that links concepts and terms within a textbook and allows the user to navigate according to own preferences. This suggests the potential utility of tools that expand such "navigational ability" outside textbooks.

Attempts at manual curriculum curation There have been at least two efforts that we are aware of, that attempts to manually create "paths" between a selected set of resources on the web — two educational start-ups, Metacademy [5], and Knewton [4]. While motivated by the same goal, we believe that manual web-scale curriculum curation is akin to the manually-curated directory of the web (not too different from the original Yahoo directory from the 1990s), i.e. offering poor scaling capability in the dynamic, growing landscape of educational content on the web.

Attempts at automatic curriculum curation Most relevant to our task is the work of [11] that attempt to infer prerequisite relationships between a pair of Wikipedia articles. They frame the problem of prerequisite prediction as "link-prediction" between a pair of pages using primarily graph-derived (e.g. hyperlink structure) and some content-derived features (e.g. article titles). In contrast to their approach, we do not assume any existing structure connecting the web resources (e.g. within Wikipedia), as the majority of the educational content on the web is unstructured. Our approach also naturally facilitates a scalable assimilation of new content, as we require only a document-scoped term-level classification, without needing to explicitly construct or update a prerequisite graph. Furthermore, we develop an approach for optimizing curricular paths using the proposed representation. More recent work of [8] develop a method that does not rely on a manual annotation of the prerequisite relations as in [11], and instead uses the statistics of concept reference in a pair of pages to determine the prerequisite relation between them. Similar to [11], their focus is on the pairwise link prediction, in contrast to our goal of globally optimizing a learning curriculum.

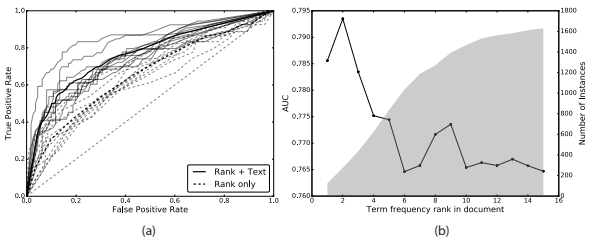


Figure 1: (a) ROC curves for the task of binary aspect classification. (b) AUC (left y-axis) of aspect classification for terms with a maximum document rank given on x-axis. Shaded region shows the number of terms up to the given maximum rank (right y-axis).

3. MODEL

3.1 Modeling explanations

We model the problem of identifying the explained and assumed terms in a document as a term-level binary classification task, i.e each term in the document is classified into one of the two categories. Although simple from an implementation perspective, this task is made difficult by the lack of annotated data in this domain. In this work, we rely on (i) manual annotation of the term aspects performed by us for one of the textbooks (Rice University's statistics text) and (ii) explicit annotations from the index of Bishop's Pattern Recognition and Machine Learning textbook that were made by the author of the text (the annotation is in the form of a location in the text where a particular concept is explained).

The Rice University's *Online Statistics Education: An Interactive Multimedia Course of Study* textbook, from hereon referred to as STATSBOOK consists of a total of 112 units, with a median of 12.5 unique technical terms per unit, for a total of 339 different technical terms in the book. We scrape the text content of the book from the web, replace all mathematical formulae and symbols with special tokens, and manually annotate each technical term mention with its representative form from the index, i.e. *normally distributed* with *normal distribution*. Manual term annotation obviates the need for introducing a word-sense disambiguation component and additional errors. We process the PRML dataset in an identical manner.

Each technical term in every unit of the book was annotated with the binary $\{explain, assume\}$ aspect, following the definitions outlined on the previous page. While for most terms, the application of these definitions is fairly unambiguous, for a significant number of term mentions, the aspects are not mutually exclusive, i.e. the term may be construed to belong to both aspects simultaneously. Often, in using (assuming) a term to explain a related concept, something about the assumed term is also explained as a side effect. The degree to which the explanation is distributed between the terms is difficult to judge objectively, and may vary between distinct mentions of the terms in different parts of the same document. We adopt a simple strategy for "breaking ties" in such cases: if we a judge a term as having been *intended* to be explained in the given context by the author, we mark it with the *explain* aspect, otherwise, the term is assumed to be *assumed*. In total across the entire STATSBOOK corpus, 1878 terms were annotated for their aspect (note that the same term appears in multiple documents with potentially different aspects), with a class ratio of 537 terms belonging to the *explain* and 1341 terms belonging to the *assume* aspect.

The PRML dataset contains a total of 3883 annotated terms, with 222 terms belonging to the *explain* and 3661 terms belonging to the *assume* aspect. The aspect of the term was determined from the index of the book, which explicitly specifies the pages where a term is explained.

A logistic regression model (LIBLINEAR [3] with default regularization parameter) was trained to predict a binary aspect of the terms and evaluated with 10-fold stratified cross-validation. A set of lexical and dependency features describing the context of each term (within a 1 sentence window), positional features describing the location of the term's mention within the document and sentences in which the term appeared, and the frequency rank of the term within the document were employed. We compare the performance of a classifier that uses all of these features with the one that uses only the rank. A classifier that is given rank as the only feature, will essentially learn a rank "threshold" that will decide the aspect of the term within the document, i.e. predict all terms above a certain rank as *explained*.

Figure 1(a) summarizes the performance of aspect prediction with the classifier trained using both linguistic and rank features (Rank+Text, AUC=0.76) versus a classifier trained using only the rank (Rank only, AUC=0.66) for the STATSBOOK corpus. As expected, rank is predictive of the aspect, but contextual linguistic cues provide a significant boost.

Keeping our end goal in mind, under Assumption 2 stated in the introduction, we hypothesize that the frequency rank of the term in a document correlates with the degree to which a term is either assumed or explained in that document. In the downstream task of linking documents to their prerequisites, getting the aspects of the more frequent terms correct is arguably more important than of the terms that only appear once or twice. We evaluate the performance of our aspect classifier as a function of the term's rank. Figure 1(b) illustrates predictive performance (AUC) on a subset of the data stratified by the term's frequency rank. We observe a favorable trend in increased predictive performance for higher ranked terms. An obvious explanation is that more frequent terms accumulate a larger set of features describing them (since each mention of the term contributes its context features), effectively decreasing variance in the predictions.

3.2 Optimal learning paths

Consider now that we have a large collection of documents (e.g. tutorials, papers, textbook chapters). Each such document explains some concepts but also assumes the reader's knowledge of other concepts (e.g. a tutorial may explain the concept of *normal distribution*, but may assume the knowledge of *probability* and *distribution*). We will now consider that we can reliably classify each term in each document into either the *Explained* or *Assumed* category. Consider that we also have a user who is interested in understanding a specific (target) document (or a set of target documents). The goal is to give a user a self-contained sequence of documents of

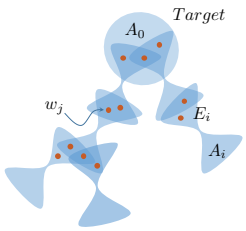


Figure 2: Each document is represented by a blue shaded region: the top part corresponds to the explained set E_i and the bottom part corresponds to the assumed set A_i. Red dots correspond to terms. This is an example of a feasible solution, where each document is *covered*.

minimal length that explains all of the concepts needed to understand the target document.

Formally each document d_i in our collection is a set of two sets of terms: the explained terms $E_i = E(d_i)$ and the assumed terms $A_i = A(d_i)$. A term in any document is either explained or assumed, but not both, i.e. $A_i \cap E_i = \emptyset$. We say that the document d_i is *covered* by a prerequisite set of documents P_i when:

$$A_i \subseteq \bigcup_{d_j \in P_i} E(d_j)$$

In other words the document is covered when every one of its assumed terms is explained by at least one document in the prerequisite set. For any prerequisite set that covers this document, the documents in the prerequisite set need to be covered as well, recursively until all documents have been covered. We assume the existence of documents with no prerequisites (leaves), i.e. those documents for which $A_\cdot = \emptyset$. The goal is to find a smallest *self-contained* set of documents P, i.e. a set of documents such that all the documents in P are covered and $d_0 \in P$, where $d_0 = \{A_0, E_0\}$ is the target document of interest to the user. Figure 2 illustrates a feasible solution to an example problem. Without additional restrictions, solutions to this problem can contain cyclical dependencies. Such cycles don't make sense in our setting. Thus an important restriction is that the the set of documents P can be ordered such that every document in the sequence is covered by the preceding documents in the sequence. Let $\mathbf{p}$ be a sequence of documents of length K, where $\mathbf{p}_k$ is the k^{th} document in the sequence, then we seek:

$$\begin{aligned} &\text{minimize } |\mathbf{p}| \\ &\text{s.t. } \forall k : A(\mathbf{p}_k) \subseteq \bigcup_{k'=0}^{k-1} E(\mathbf{p}_{k'}) \\ &d_0 \in \mathbf{p} \end{aligned} \tag{1}$$

ILP formulation

We formulate an Integer Linear Program (ILP) that finds a minimum length self-contained sequence $\mathbf{p}$ of at most K documents such that it covers a user's document of interest d_0. Consider that we have a total of D documents. We define the following variables:

$$x_i^k \in \{0,1\} \quad \text{document } d_i \text{ is in } k^{th} \text{ position in the sequence}$$

We define the following constants:

$$\begin{aligned} e_{ij} &\in \{0,1\} \quad \text{Term } j \text{ is explained in document } i \\ a_{ij} &\in \{0,1\} \quad \text{Term } j \text{ is assumed in document } i \end{aligned}$$

Each assumed term in a document in position k must be explained by at least one document up to (but not including) the document in position k. This can be expressed via the following constraint:

$$\sum_{k'}^{k-1} \sum_{i}^{D} e_{ij} x_i^{k'} \geq \sum_{i}^{D} a_{ij} x_i^k \ \forall j \forall k$$

Each position in the sequence contains at most 1 document:

$$\sum_{i}^{D} x_i^k \leq 1 \ \forall k$$

User's preference of covering a document of interest d_0 is an additional constraint:

$$\sum_{k}^{K} x_0^k = 1 \ \forall k$$

Finally, the objective is to minimize the number of documents in the sequence:

$$\text{minimize } \sum_{k}^{K} \sum_{i}^{D} x_i^k$$

The above formulation also allows us to directly incorporate the user's prior knowledge into this optimization problem. If we represent a user as a set of *explained* terms, i.e. terms that the user is assumed to have mastered, then the constraints corresponding to these terms may simply be dropped from the formulation.

In the most general case, this formulation has D^2 variables and $O(D^2 \times V)$ constraints, where V is the number of terms in the vocabulary. In practice, however, we will often limit the maximum allowable sequence length to a fairly small constant (e.g. 10, as done in our experiments), reducing the order of the problem to $O(D)$ variables and $O(D \times V)$ constraints.

While in extremely large settings (hundreds of thousands of documents), even with a small K, solving this ILP directly is infeasible, in practice, we find that that we can can obtain exact solutions using LP relaxation and a vanilla Branch and Bound (using GLPK[1]) within several seconds, even with a many as 1,000 documents and hundreds of terms. Developing an approximation algorithm based on rounding the LP solution is our ongoing work.

[1] `https://www.gnu.org/software/glpk/`

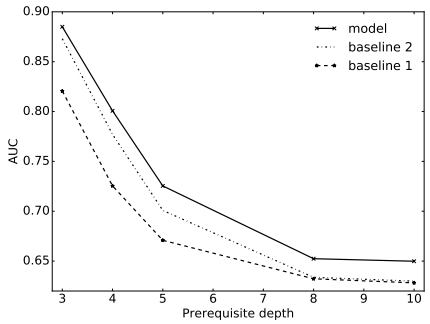


Figure 3: Term aspect classification is useful at the task of recovering prerequisites for units within a textbook. The y-axis is the average AUC at the task of predicting whether a particular unit is a prerequisite of another unit, based on three metrics. The metric that incorporates the *Explain/Assume* classifier performs best (solid line).

4. EVALUATION

4.1 Prerequisites

In order to evaluate the *Explain/Assume* classifier in an end-to-end setting, we employ the output of this classifier in the task of predicting prerequisites in a dataset where the prerequisites have been explicitly annotated. One such resource is *Rice University's Online Statistics Textbook*, which in addition to the text content, provides an explicit dependency graph annotating prerequisite relations between pairs of units (units are at the level of chapter sections). We propose a metric for scoring a pair of units according to their prerequisite relationship based only on the terminology of both units and the output of the *Explain/Assume* classifier. The proposed "prerequisite score" is defined as follows:

$$P(d_a \rightarrow d_b) = \frac{\sum_{t_i \in d_b} n_i^a \mathbb{1}[t_i \text{ assumed in } d_b \wedge \text{ explained in } d_a]}{\sum_{t_j \in d_a} n_j^a \mathbb{1}[t_j \text{ explained in } d_a]}$$

where n_i^a is the number of occurrences of term i in document d_a. Since the above score is guaranteed to be in the $[0, 1]$ range, we can interpret it as a probability $P(d_a \rightarrow d_b)$, a probability that document a is a prerequisite of document b. There is an intuitive interpretation to the above score: a document can be considered a strong prerequisite of a target document when it explains all of the assumed terms in the target document and nothing more. We can convince ourselves that in this case the score as defined above will be equal to 1. A document that explains too many unrelated concepts will suffer a penalty with respect to its prerequisite score to another document. Furthermore, we consider the relative frequency of the explained term in the prerequisite document as an additional signal of that term's importance. We find that this additional information increases the performance of prerequisite classification (discussed at the end of this section).

Because the output of the *Explain/Assume* classifier is a probability, rather than a class, we can relax the above score to directly incorporate the uncertainty in the classification:

$$P(d_a \rightarrow d_b) = \sum_{t_i \in d_b} \frac{n_i^a P(t_i \text{ explained in } d_a)}{\sum_{t_j \in d_a} n_j^a P(t_j \text{ explained in } d_a)} \quad (2)$$

Note that in addition to relaxing the requirement of an explicit *Explain* or *Assume* label, we also drop the requirement that only the assumed terms need to be explained to count towards the prerequisite score. This distinction is optional, but it encodes an important assumption on the kinds of "prerequisites" that this score will discover. This also brings up the importance of being precise about the definition of a prerequisite. A document a is a strict prerequisite of document b, if document a explains a subset of the assumptions in document b. However, we can relax this definition by *not* requiring that the terms explained in the prerequisite (a) are strictly assumed in the target (b). In other words, a document that explains a subset of the terms also explained in the target and *nothing else*, will have a score of 1 according to the above equation. In practice this corresponds to documents that explain the same concepts but in a simpler way (since they explain only a subset of the explained concepts in the target), and this is often a desired behavior in a learning sequence. For example, before reading a more advanced article on *Support Vector Machines*, the learner might want to read a more basic introduction to *Support Vector Machines*, although from the perspective of term classifications, both documents explain the same concept.

4.1.1 Reconstructing prerequisites

Rice University's Online Statistics Textbook provides a valuable resource for evaluating the effectiveness of the *Explain/Assume* classification at the task of predicting prerequisite relations between documents. The textbook consists of 112 units at the granularity of chapter sections, annotated as a directed graph, i.e. specifying a directed edge between a pair of units if one unit is considered a prerequisite of another unit. We process the raw HTML files of the textbook by removing markup, segmenting sentences and extracting terminology (obtained from the index) features as described in Section 3.1. We pose the problem of prerequisite relation prediction as a standard binary classification task, i.e. predicting for each pair of units in the book whether one unit is a prerequisite of another, where we consider a pair of units to be in a gold-standard prerequisite relation if there is a directed path between them in the graph. AUC is a convenient metric for evaluating performance in this prediction task, as the output of our scoring metric (Equation 2) is already scaled between 0 and 1. Note that the model trained only on the PRML corpus was used for term-aspect classification in this task. Figure 3 illustrates the results for three different models, as a function of the prerequisite depth, i.e. stratifying the classification results for a pair of units by the maximum distance between them in the graph. The three models evaluated are as follows:

- **Model** Prerequisite score is computed with Equation 2.
- **Baseline 1** Prerequisite score is computed with Equation 2, but with all n_i^a, n_j^a and $P(t_\cdot \text{ explained in } \cdot)$ set to 1. This baseline is equivalent to a ratio between the number of overlapping terms between a pair of documents and the number of terms in the prerequisite, i.e. $\frac{|d_a \cap d_b|}{|d_a|}$.

- **Baseline 2** Prerequisite score is computed with Equation 2, but with $P(t.\ \text{explained in}\ \cdot)$ set to 1.

Each baseline illustrates the effect of *not* including a component of the scoring function in Equation 2. Our first conclusion from the results in Figure 3 is that the output of the *Explain/Assume* classifier provides an important signal in predicting the prerequisite relationship between documents. Furthermore, the relative frequency of the explained terms in the prerequisite document provides an additional gain in performance. This can be explained by Figure 1(b): the performance of the *Explain/Assume* classifier is greater in the higher term-frequency regime; discounting low-frequency terms (that are also likely less important to the content) reduces the classification noise and boosts the performance at the prerequisite prediction task. An additional observation is that the performance of the pairwise prerequisite classification improves for pairs of units that are closer in the graph, i.e. with less units in between. This is easily explained: units that are farther apart typically share less terminology, making the estimates based on terminology overlap noisier.

It is also interesting to note that the simplest baseline that considers only the ratio of overlapping terms between a pair of documents to the total number of terms in the prerequisite document does surprisingly well, especially well for pairs of documents closer together. This can be explained as follows: in a sequence of units like those in a textbook, units that are prerequisites tend to be less advanced, i.e. have less terminology, since less of it was introduced up to that point. Thus, units that are prerequisites, at least in a textbook, would be fairly predictable from the relative frequency of overlapped terms alone.

4.2 Scaling to the web

We collect and release two web corpora of educational content in the areas of Machine Learning and Statistics. Both corpora were collected using Bing Search API, by querying for short permutations of terms collected from the index of the *Pattern Recognition and Machine Learning* and *Rice University's Online Statistics Textbook*. The two corpora contain 42,000 and 1,000 documents respectively – a mixture of HTML and PDF files, pre-processed and converted to plain text. The difference in size of the two corpora is due to a smaller set of keywords used in the query set, and used primarily to rapidly validate the proposed model for path optimization. Consequently, because of a smaller term vocabulary, the smaller corpus is significantly less noisy (less irrelevant documents). The union of the terminology from the index of both textbooks was used as the vocabulary in processing each document. Additionally, terminology variations and abbreviations were consolidated using the link data from Wikipedia, e.g. terms *EM*, *E-M*, *Expectation-Maximization*, are all mapped to the same concept of *EM* in the terminology extraction stage.

Following the extraction of terminology from each webpage, each term is classified using the *Explain/Assume* classifier trained on the *Pattern Recognition and Machine Learning* textbook. We train this classifier in a fully supervised setting using all of the annotated data. In the next several sections, we present the analysis of the two web corpora and demonstrate the effectiveness of the proposed approach to connecting educational resources on the web.

4.3 Diversity of assumptions

The web is a unique setting, that unlike a traditional textbook or a course, offers a multitude of diverse explanations of the same concept. This diversity potentially enables the level of personalization that is not possible in traditional resources. We can analyze the diversity in the educational content on the web by looking at a slice of the web resources that share the same topic, but differ in their underlying assumptions and explanations. Figure 4 illustrates two articles that are both on the topic of *Expectation Maximization*. However, the two articles differ significantly in their assumptions on the background of the reader. Article 1 (left in Figure 4) is a very basic introduction to the topic and does not assume the knowledge of even the concept of *maximum likelihood*, which under most traditional curricula is assumed to be the prerequisite. Article 2 (right in Figure 4), however, assumes the knowledge of many more concepts such as *posterior probability*, *likelihood function* and *maximum likelihood*. This difference in the distribution of the underlying assumptions is explained by the fact the Article 1 s a very basic introduction to the topic, intended for an audience not in the area of statistics or machine learning. Article 2, however, is a significantly more thorough and a more technical introduction to the concept of the *Expectation Maximization* algorithm and thus assumes significantly more prerequisite background in the areas of statistics and machine learning. It's important to note that this distinction between the two documents cannot be easily made from their titles, or other surface cues: both documents are approximately the same length and their titles do not give away the level of technical detail. Their text content, however, provides the necessary cues to this information.

4.4 Fundamental prerequisites

Figure 5 illustrates the result of optimizing a learning path over the web corpus of 1,000 documents for the target webpage on the topic of "Maximum Likelihood Estimation". Sequences were optimized using the ILP formulation described in Section 3.2 using the GLPK Branch and Bound solver. Red rectangles correspond to terms for which the predicted label is *assumed* in the given document, and blue otherwise. In addition to the term-coverage diagram, we also illustrate the prerequisite dependencies extracted from the term coverage data: a directed edge is drawn to a document from the closest prerequisite in the sequence that covers at least one assumed term in the document. In the example in Figure 5, the target web-page is a fairly technical article on *Maximum Likelihood Estimation* that assumes the reader's understanding of the concepts such as the *likelihood function* which is pivotal for understanding the concept of *maximum likelihood*. As a consequence, the web-page that is placed immediately before in the optimal sequence are slides which consist of a more basic introduction to the *maximum likelihood*. Furthermore, the original target article assumes the reader's familiarity with *Generalized Linear Models* (which is in fact the previous section of the lecture notes of that series, indicating it as a prerequisite). The resulting sequence also contains an additional prerequisite on this topic. Finally, an interesting observation is that while the target article is fairly advanced in its assumptions about the reader's knowledge of

Figure 4: An example of two different web-pages about the same topic: *Expectation Maximization*, together with each page's terminology and its classification into either the *Explained* class (green) or the *Assumed* class (red). Observe that the two pages, while about the same topic, are different in what they assume about the reader. The article on the left is a very basic introduction to this topic, while the article on the right is written for experts.

probability, it actually goes into surprising depth in explaining the concept of a *derivative* and maximizing a function using derivatives from scratch, which is another important prerequisite to the concept of *maximum likelihood*. This is highly unconventional in traditional textbook and course curricula. This again underlines the advantage of working with the assumptions at document-level, allowing to leverage the diversity in explanations to find "shortcuts" through the learning paths.

Figure 6 provides additional insightful examples of the generated sequences extracted from the term-coverage data of each sequence. Figure 6(d) is another example where the target document is a fairly advanced introduction to the topic (*Expectation Maximization*), which is preceded by a more gentle introduction to the same topic, as well as an additional prerequisite (*Maximum Likelihood*) which is a common prerequisite for this topic. Note, however, that while *Maximum Likelihood* is traditionally considered as a prerequisite for learning about *Expectation Maximization*, it is not the case for the more basic introduction to this topic (*What is the Expectation Maximization algorithm*), as that particular introduction aims to bring a very high-level understanding of the topic without burdening the reader with additional prerequisite requirements. Therefore, in that particular sequence, the reader is first given a gentle introduction to the topic, then the necessary prerequisite (*Maximum Likelihood*) for understanding the more advanced introduction.

4.4.1 Error analysis

The extracted sequences are not without errors. These errors stem from several potential sources, as a fairly involved pipeline lies between the raw document and the resulting optimal sequence, providing an opportunity for errors to propagate through the different stages. We break down these errors by their source to give a better understanding of how these problems need to be addressed in future work:

Terminology extraction: The greatest source of errors stems from errors in terminology extraction. There are two types of errors involved in terminology extraction: *false negatives* (missing terms) and *false positives* (term sense disambiguation errors). False negatives are more difficult to detect and often result in missing prerequisites; missing terms are especially difficult when relying on a finite vocabulary.

***Explain/Assume* classification:** The second greatest source of errors are the mistakes made by the aspect classifier. Classifying an explained term as an assumed term creates unnecessary prerequisites, while the reverse results in missing potentially important prerequisites.

Path optimization: because we solve the optimization problem exactly (i.e. find a global optimum), there are no errors stemming from the optimization itself (this will become a potential source of errors, however, when an approximation scheme, e.g. LP rounding, is used to obtain an approximate solution). However, the formulation of the optimization problem can be improved so as to introduce robustness to the errors in the earlier stages of the pipeline. As path optimization is the final stage that produces the final output, its sensitivity to the errors in terminology extraction and term aspect classification are directly reflected in the resulting output. Introducing robustness to these errors directly in the formulation of the optimization problem is potentially the most effective way to address the issues in the earlier stages of the pipeline. One issue with the current formulation is its inability to incorporate the relative frequency of the term into the optimization objective: ideally terms that appear less frequently in a document should have a lesser precedence for coverage than those that appear more frequently (Assumption 2 in the Introduction). The example in Figure 5 demonstrates the lack of robustness in the third document, where the appearance of the term *integral* creates an additional sequence of documents that cover this concept. From our earlier analysis is Section 3.1, we have shown that the errors in the *Explain/Assume* classifier are directly related to the relative frequency of the terms, and thus a way to incorporate these frequencies as weights into the optimization would potentially be the most effective way to deal with this noise.

5. CONCLUSION

We developed what we believe is the first end-to-end approach towards automatic *curriculum extraction* from the web, relying on the following pipeline: 1) extracting what is assumed vs. what is explained in a single document and then 2) connecting these documents into a sequence ensuring that the progression builds up the knowledge of the learner gradually towards their goal. We developed algorithms that addressed both of these components: 1) a semi-supervised approach for learning a term aspect classifier from a very small set of annotated examples and 2) an optimization problem for learning path recommendation based on the user's learning goals. To the best of our knowledge, we for the first time demonstrate and leverage the most unique characteristic of the web in the domain of learning: *diversity*, i.e.

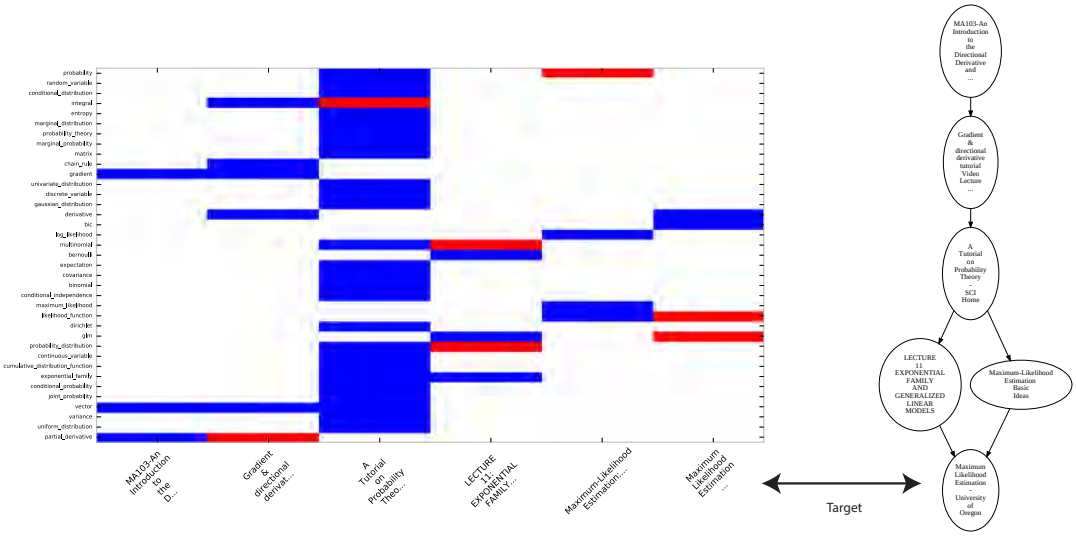


Figure 5: An example optimal sequence for the target document on *Maximum Likelihood Estimation*. Left: the term-coverage diagram. Each column represents a single web-page and each row a single term. Red rectangles correspond to terms that are classified as *assumed* in the corresponding document and *blue* corresponds to the *explained terms*. Right: the term-cover diagram is converted into a directed graph whereby an edge is drawn to a document from its closest prerequisite that explains at least one assumed term.

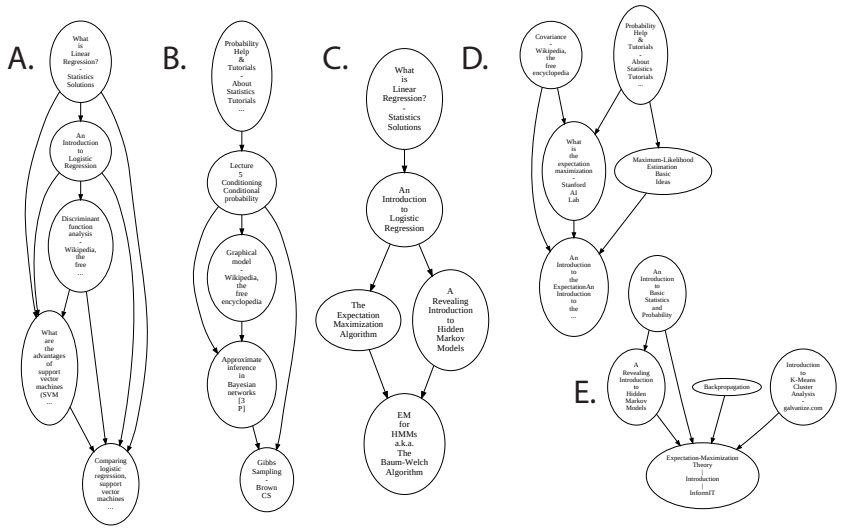


Figure 6: Additional examples of optimal paths generated from the 1,000-document web-page corpus for a select set of target web-pages. See text for details.

presence of content that explains the same concepts but in many different ways and from many different angles. This property of the web opens the doors to personalizing learning sequences that leverage the differences in explanations to find the most effective paths and shortcuts through the Internet. Finally, we outlined a set of important challenges that need to be addressed in order to make this task a practical reality at web-scale. We hope that this work, in addition to the datasets that we release, will serve to inspire interest from the community in what we believe is a challenging and an important task.

Acknowledgements

This research was funded by a grant from the John Templeton Foundation provided through the Metaknowledge Network at the University of Chicago. Computational resources were provided in part by grants from Amazon and Microsoft.

6. REFERENCES

[1] R. Agrawal, S. Chakraborty, S. Gollapudi, A. Kannan, and K. Kenthapadi. Empowering authors to diagnose comprehension burden in textbooks. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 967–975. ACM, 2012.

[2] R. Agrawal, S. Gollapudi, A. Kannan, and K. Kenthapadi. Study navigator: An algorithmically generated aid for learning from electronic textbooks. JEDM-Journal of Educational Data Mining, 6(1):53–75, 2014.

[3] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. JLMR, 9:1871–1874, 2008.

[4] J. Ferreira. Knewton. `http://http://www.knewton.org`.

[5] R. Grosse. Metacademy. `http://www.metacademy.org`.

[6] B. Jones, E. Reedy, and B. A. Weinberg. Age and scientific genius. Technical report, National Bureau of Economic Research, 2014.

[7] B. F. Jones. As science evolves, how can science policy? In Innovation Policy and the Economy, Volume 11, pages 103–131. University of Chicago Press, 2011.

[8] C. Liang, Z. Wu, W. Huang, and C. L. Giles. Measuring prerequisite relations among concepts.

[9] M. Peat, C. E. Taylor, and S. Franklin. Re-engineering of undergraduate science curricula to emphasise development of lifelong learning skills. Innovations in Education and Teaching International, 42(2):135–146, 2005.

[10] D. J. Rowley, H. D. Lujan, and M. G. Dolence. Strategic Choices for the Academy: How Demand for Lifelong Learning Will Re-Create Higher Education. The Jossey-Bass Higher and Adult Education Series. ERIC, 1998.

[11] P. P. Talukdar and W. W. Cohen. Crowdsourced comprehension: predicting prerequisite structure in wikipedia. In Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, pages 307–315. Association for Computational Linguistics, 2012.

[12] H. Tyson-Bernstein. A conspiracy of good intentions. america's textbook fiasco. 1988.

[13] S. M. Wilson and P. L. Peterson. Theories of learning and teaching: what do they mean for educators? National Education Association Washington, DC, 2006.

Calibrated Self-Assessment

Igor Labutov
Cornell University
iil4@cornell.edu

Christoph Studer
Cornell University
studer@cornell.edu

ABSTRACT

Peer-grading is widely believed to be an inexpensive and scalable way to assess students in large classroom settings. In this paper, we propose *calibrated self-grading* as a more efficient alternative to peer grading. For self-grading, students assign themselves a grade that they think they deserve via an incentive-compatible mechanism that elicits maximally truthful judgements of performance. We show that the students' self-evaluation scores obtained via this mechanism can be used to perform classic item response theory (IRT) analysis. In order to obtain unbiased estimates of the IRT parameters, we show that the self-assigned grades can be calibrated with a minimum amount of input from instructors or domain experts. We demonstrate the effectiveness of the proposed calibrated self-grading approach via simulations and experiments on Amazon's Mechanical Turk.

Keywords

Assessment, self-grading, item response theory (IRT).

1. INTRODUCTION

A significant bottleneck in scaling traditional classrooms to hundreds or thousands of students is the challenge of enabling efficient mechanisms of assessment. Peer-grading, hailed as a solution to this "scaling problem," has received significant attention, both from the education [12, 5] and machine learning [10, 11] communities. Broadly speaking, peer-grading can be thought of as a relaxation of the traditional teacher/student roles in the classroom: An expert instructor is replaced by several "noisy" students having the task of estimating performance of other students. Virtually all of the existing statistical models for peer-grading aim to estimate the student's true performance from such noisy measurements, under some metric of optimality.

Self-grading constitutes a special case of peer-grading: The student is their own only "peer" and is solely responsible for assigning a score based on the judgement of their own work. Depending on the student's honesty in self-evaluation, self-grading is appealing for at least two reasons: (i) Students can provide a richer signal towards their internal state of knowledge by explicitly revealing confidence in their answers—a signal that can be exploited during assessment; (ii) because every student is their own grader, potentially no additional peer-grading efforts are required to perform assessment. Self-grading, however, introduces two unique challenges not faced in traditional peer-grading: (i) Designing mechanisms for eliciting honest judgement of performance and (ii) accounting for individual biases in self-evaluation. The first challenge in self-grading fundamentally requires an explicit mechanisms for eliciting truthful judgements.[1] The second challenge is addressed in peer-grading by appealing to statistics and assuming that the population of graders is—at least on average—unbiased.

In this work, we propose *calibrated self-assessment* to address both of the above challenges. Our approach combines self-assessment with a small number of instructor-graded items, which provides a simple, incentive-compatible mechanism of eliciting self-assigned scores, and yields assessments of comparable or superior quality to a setting with significantly more instructor-graded items and no self-scoring. As a consequence, calibrated self-assessment enables a significant reduction in effort of instructors, domain experts, or peers.

2. RELATED WORK

We focus our review on two research directions that our work aims to bring together: (i) self-assessment as a method for summative assessment and (ii) decision-theoretic mechanism design for judgement elicitation.

Self-grading and Peer-grading in education: Self-assessment is often seen by teachers as a valuable tool in classrooms [17], who cite self-assessment as a viable way to reduce the instructor's effort, elicit additional information from students (e.g., their effort and confidence), and provide an additional learning opportunity in the process. More recently, in addition to peer-grading, self-grading was deployed in massive open online courses (MOOCs) [5]. Self-grading as a tool for summative assessment, however, is controversial, with its validity questioned on the basis of students' internal biases. In fact, studies indicate that bias is often a function of one's ability [17, 16]. Studies that compare peer-grading and self-grading differ in their findings, with self-grading and

[1]This is also a potential problem in peer-grading when conflicts of interest are present.

peer-grading performance excelling in different conditions (classrooms, age-groups, etc.), but both are heavily influenced by the underlying assessor biases (see [16] for a survey of the studies). A study carried out in four middle-school science classrooms found that peer-grading and self-grading have a high correlation with instructor grades, with grading bias patterns that are consistent with other studies [12]. In addition, they found that the process of self-grading resulted in learning gains, whereas peer-grading did not. A recent study carried out at the university level, however, found that both peer-grading and self-grading results in learning gains as a side-effect of grading [8].

The existing literature on self-grading points to the significant effect of bias in self-scoring, with most studies concluding that students of lower ability tend to inflate their grades more. As a consequence, we argue for the importance of an incentive-compatible mechanism that is designed to elicit maximally truthful judgements, and a *calibrated* model that is able to explicitly de-bias the individuals by incorporating a subset of instructor-graded items.

Judgement elicitation: The literature on truthful judgement elicitation through scoring functions dates back to the fifties, when the so-called "quadratic scoring rule" was proposed for the task of weather forecasting [2]. Since then, a number of generalizations of the quadratic scoring rule and other incentive-compatible scoring rules have been proposed and analyzed [3, 14, 7, 13] and found application in forecasting weather, sports, and finance. Analysis of the behavior of non-risk neutral agents in scoring-rule-based mechanisms has received only limited attention [9], with lottery-based payoffs being the most well-known solution for encouraging risk-neutral behavior. Lottery-based payoffs had received mixed results in experimental evaluations [4, 15], and in the context of education a reward system based on a lottery is not a reasonable solution. In this work, we rely on heavily limited instructor input in order to correct for individual biases, which includes under- and over-confidence, as well as non-risk-neutral behavior.

To the best of our knowledge, the only work that applies a scoring rule mechanism in the context of education that we are aware of is [1]. The focus of this work is in analyzing the effect of different scoring functions on the self-assessment behavior of students. Our primary contribution in this work is in developing a principled statistical model for calibrated summative assessment that integrates self-scoring and instructor-scoring within the classic IRT framework.

3. MODEL

Self-grading without a proper incentive mechanism may lead to dishonest behavior. In the setting of self-grading, a "mechanism" is a *scoring rule* that specifies the rules by which the points are assigned to the student as a function of their own judgement and the outcome (i.e., whether their answer was correct). A mechanism is called *incentive compatible* when the student's optimal strategy with respect to his or her own utility function results in a truthful elicitation of information, e.g., truthful judgement of their own work.

We consider the following scoring function:

$$p_{ij} = \begin{cases} \theta_{ij} & \text{if correct} \\ -\frac{1}{2}\theta_{ij}^2 & \text{if wrong,} \end{cases}$$

where $\theta_{ij} \in [0, A]$ is a score provided by student i in answering question j, where A is some fixed upper bound. If the student provides a correct answer, they get the θ_{ij} points that they proposed; if they provide an incorrect answer, they lose exactly half of that value squared. This scoring function is known as a *quadratic scoring rule* and was first proposed in [2].

For this scoring function, the expected payoff is

$$\mathbb{E}[p_{ij}] = \theta_{ij}\hat{\pi}_{ij} - \frac{1}{2}\theta_{ij}^2(1 - \hat{\pi}_{ij}), \tag{1}$$

where $\hat{\pi}_{ij}$ is the i^{th} student's estimate of the probability that they will get question j correct. This expression is maximized when

$$\theta_{ij} = \frac{\hat{\pi}_{ij}}{1 - \hat{\pi}_{ij}}. \tag{2}$$

Equation 2 is exactly the student's own belief about the odds of them answering the question correctly. Consider that the student estimates their chances of answering any question correctly, by simultaneously estimating their own ability and the difficulty of the question. Let us now define that probability to be the standard IRT Rasch likelihood, but defined with respect to the student's own estimate of their ability, $\hat{s}_i$ and their estimate of the question's difficulty $\hat{q}_j$:

$$\hat{\pi}_{ij} = \frac{1}{1 + \exp(-(\hat{s}_i - \hat{q}_j))}.$$

Given the student's estimate of their own ability $\hat{s}_i$ and of the difficulty of the question $\hat{q}_j$, we can now derive their optimal proposed score (assuming they act rationally and are risk-neutral) for that problem θ_{ij} (or rather its logarithm):

$$\log(\theta_{ij}) = \hat{s}_i - \hat{q}_j,$$

which follows from the fact that log-odds of a logistic model is a linear function of its parameters. We will assume that the student is risk-neutral and is unbiased in his or her estimates of own ability and question difficulty, but we will relax both assumptions later. On any given question, however, the student's estimate of their ability to answer that particular question may deviate from their true ability. Assuming that the student's own estimates are normally distributed around their true values, we get:

$$\hat{s}_i - \hat{q}_j \sim \mathcal{N}(s_i - q_j, \sigma^2),$$

where s_i and q_j are the true student ability and question difficulty parameters respectively. As a consequence, it follows that $\log(\theta_{ij})$ is normal distributed and θ_{ij} is log-normal distributed. Consider a dataset D consisting of the self-assigned scores $\log(\theta_{ij})$ submitted by each student for each question that the student answered. We can write the conditional likelihood of the entire dataset as follows:

$$P(\boldsymbol{\theta} \mid \mathbf{s}, \mathbf{q}) = \prod_{(i,j)\in D} \mathcal{N}(\log(\theta_{ij}) \mid \mu = s_i - q_j, \sigma^2).$$

Here, $\mathbf{s}$ and $\mathbf{q}$ are the vectors comprising the student ability and question difficulty parameters, respectively, and $\boldsymbol{\theta}$ is the vector of student-submitted scores. Maximizing the

likelihood of all observations gives a straightforward least-squares solution for the parameters s_i and q_j, given all the user-provided scores θ_{ij}. Note that σ^2 is assumed to be a constant variance in students' estimates of their own ability. In practice this variance is likely user-specific and corresponds to the students' ability in self-assessment. We will address the issues of bias and variance in self-assessment in Section 3.2.

3.1 Parameter estimation

It is interesting to note that we can solve for the IRT parameters (student abilities and question difficulties) using the above formulation with *no* outcome information, i.e., without knowing which students answered which questions correctly. In fact, the above approach does not even require that the students who are self-grading know what the correct answer is; students' confidence in their answers elicited through the quadratic scoring rule is all that is needed to learn the parameters of the model. Of course, this observations relies on two fundamental assumptions: (i) students are risk-neutral and (i) students are unbiased in estimating their chance of answering a question correctly. In Section 3.2, we will account for the individual biases and non-risk-neutral behavior by explicitly introducing a bias parameter into the model and estimating it from an additional set of instructor-graded responses. However, in order to gain a better understanding of the model, it is insightful to first analyze the solution to the problem where both of these assumptions hold.

The solution for the model parameters can be obtained in closed-form using a standard pseudo-inverse solution to a least-squares problem. Alternatively, the solution can be obtained iteratively, without requiring to explicitly invert any (potentially large) matrices. In particular, one can repeatedly evaluate the following two steps:

$$s_i = \sum_{j \in Q_i} \frac{q_j}{\lambda + n_q^i} + \sum_{j \in Q_i} \frac{\log(\theta_{ij})}{\lambda + n_q^i}$$
$$q_j = \sum_{s \in S_j} \frac{s_i}{\lambda + n_s^j} - \sum_{i \in S_j} \frac{\log(\theta_{ij})}{\lambda + n_s^j}.$$

Here, s_i is the ability of student i and q_j is the difficulty of question j. To guarantee a unique solution, we introduce a non-negative regularization parameter λ, which we will discuss in more detail in the next paragraph. The constants n_q^i and n_s^j are the number of questions that student i answered and the number of students that answered question j respectively. Note that the above iterative solution has an intuitive interpretation: The ability of the student is the sum of the average of the (log-transformed) self-assigned scores to a set of questions that the student answered and the average difficulty of those questions. In turn, the difficulty of a question is the negative of the average (log-transformed) score that students assigned to themselves for that question plus the average ability of the students who answered that question. Intuitively, if students with high ability self-assess themselves to have done poorly on a specific question, that question will have a large difficulty parameter.

In the case where there is no missing data, i.e., each student answers each question, the solution for student ability parameters simplifies to:

$$\mathbf{s} = \begin{bmatrix} \frac{\sum_{i \in S} \log \theta_{i1}}{\lambda + N_s} \\ \vdots \\ \frac{\sum_{i \in S} \log \theta_{iN_q}}{\lambda + N_s} \end{bmatrix} + \mathcal{O}(1/\lambda)\mathbf{1},$$

where $\mathcal{O}(1/\lambda)$ is a function that grows proportional to $1/\lambda$. In other words, the student's ability is simply the average of the (log-transformed) scores that the student assigned to themselves plus a constant that is identical for each student. This solution also illustrates the role of the regularization parameter λ. Because the solution for $\mathbf{s}$ and $\mathbf{q}$ is location-invariant, without an explicit prior, the likelihood is maximized by scaling all parameters to infinity. This is equivalent to setting λ to 0, in which case the above solution will tend to infinity, as expected. Note, however, that the relative ranking of the student abilities in this solution will be consistent, regardless of λ. As obtaining the ranking of the students is our primary focus, we can thus set λ to zero in the above solution, and simply consider the average self-assigned (log-transformed) score as the the ability parameter of the student. The same argument applies to question difficulty parameters.

3.2 Calibrating the model

There are two issues in relying on students' self-given score for ranking students via the IRT model: (i) Students may be prone to over- or under-estimating their ability and (ii) because there is uncertainty involved in both answering and grading, some students may be more or less inclined to "gamble" with their self-assigned score (i.e., some students are more or less risk-averse/risk-loving). We subsume both effects (as it is impossible to tell them apart) into a general student "bias" in self-grading, and model it explicitly as

$$\log(\theta_{ij}) = \hat{s}_i - \hat{q}_j + b_i,$$

where $b_i \in (-\infty, \infty)$ is a student-specific bias. We assume that this student bias is drawn from a normal distribution $b_i \sim \mathcal{N}(0, \sigma_b^2)$, where the above distribution stipulates that the average of the student population is unbiased. It is impossible to estimate b_i using self-grading alone, as without actual observations of correctness of students' responses, the model will conflate s_i and b_i into a single parameter. Imagine that we do grade a student's responses on a small subset of the answered questions (which they also self-grade). Let the set of instructor-graded questions be $Q_g \subseteq Q$, where Q is the set of all questions. As the observations of instructor- and self-assigned grades are all conditionally independent given the student and question parameters, the overall likelihood of both self- and instructor-given scores is a product of these likelihoods. We can then express the log-likelihood of the entire dataset as a sum of the self-graded response log-likelihoods and instructor-graded response log-likelihoods:

$$\log P(\boldsymbol{\theta}, \boldsymbol{y} \mid \mathbf{s}, \mathbf{q}, \mathbf{b}) = \sum_{s_i \in S} \Bigg(\underbrace{\sum_{q_j \in Q} (\log \theta_{ij} - (s_i + b_i - q_j))^2}_{\text{self-graded responses}} + \underbrace{\sum_{q'_j \in Q_g} \log(1 + \exp(-y_{ij}(s_i - q'_j)))}_{\text{instructor-graded responses}} \Bigg).$$

Here, $y_{ij} \in \{-1, 1\}$ is the instructor-grade for question j answered by student i and $\boldsymbol{y}$ is the response vector for all students ($y_{ij} = +1$ corresponds to a correct response and $y_{ij} = -1$ otherwise). Observe that the "bias" parameter only appears in the self-graded part of the likelihood. This allows us to calibrate the model via instructor-graded questions as a "training set" to separate the effects of the bias and true ability. Note that, unlike in the previous case that relied entirely on students' self-scores, like with the traditional Rasch IRT model, we are unaware of a closed form solution for this formulation. In all of our experiments, we use the L-BFGS algorithm [18] for learning model parameters.

3.3 Consequences of students' awareness of the mechanism

The assumption that the learner is optimizing a utility function based on the expected test score:

$$\mathbb{E}[p_{ij}] = \theta_{ij}\hat{\pi}_{ij} - \frac{1}{2}\theta_{ij}^2(1 - \hat{\pi}_{ij}) \tag{3}$$

fundamentally assumes that the student believes that each question will be graded, as otherwise there would be no possibility of getting a question wrong and losing points. In practice, our goal for self-grading may be motivated by the effort to reduce the instructor's involvement in grading, and, in general, as a way to scale assessment to potentially very large classrooms, such as massive open online courses (MOOCs). Having each submission be graded by an instructor (or your peers) defeats the purpose of self-grading. If, however, the student is aware of the fact that not every question is graded, we can expect that their utility function, and thus their optimal strategy, will be affected by this knowledge. If the test is administered once, of course, the students could be deceived into believing that every question is graded. In a real course, however, a more realistic assumption is that the students possess the knowledge that not all of the questions are graded and if the assignments are returned, we can expect that the students' estimates of the fraction of graded questions will improve over time. If, however, the student believes that a random subset of their submissions is graded by someone else, but if the student does not know which subset is graded, then we should still expect the student's optimal behavior to be maximizing a utility function similar to the one above. The utility function will not be the same, as we now have to account for the student's belief about how many problems are graded by someone else. Let us assume that the student has a prior belief that each problem has a probability ρ of being graded. Then, the expected score the student i receives on question j is given by

$$\mathbb{E}_{gr}\left[\mathbb{E}[p_{ij} \mid \text{graded}]\right] = \rho(\theta_{ij}\hat{\pi}_{ij} - \frac{1}{2}\theta_{ij}^2(1 - \hat{\pi}_{ij})) + (1 - \rho)\theta_{ij},$$

where we take an additional expectation with respect to the student's belief that the problem is graded. Note that when a problem is *not* graded, the expected score that the student receives is just θ_{ij}, i.e., their self-assigned score, regardless of whether the student answers correctly. This is because when a problem is not graded, there is no possibility of losing points. We can show that the student's optimal self-assigned score $\log(\theta_{ij})$ has the following approximate relationship to their ability and question difficulty (the approximation is a piece-wise linear approximation to the true strategy that is asymptotically accurate):

$$\log(\theta_{ij}) = \max\left\{\log\left(\frac{1}{\rho} - 1\right), (s_i - q_j) - \log\rho\right\}.$$

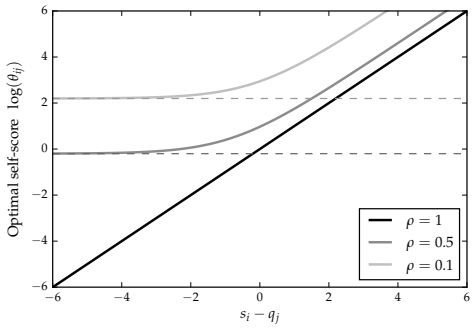


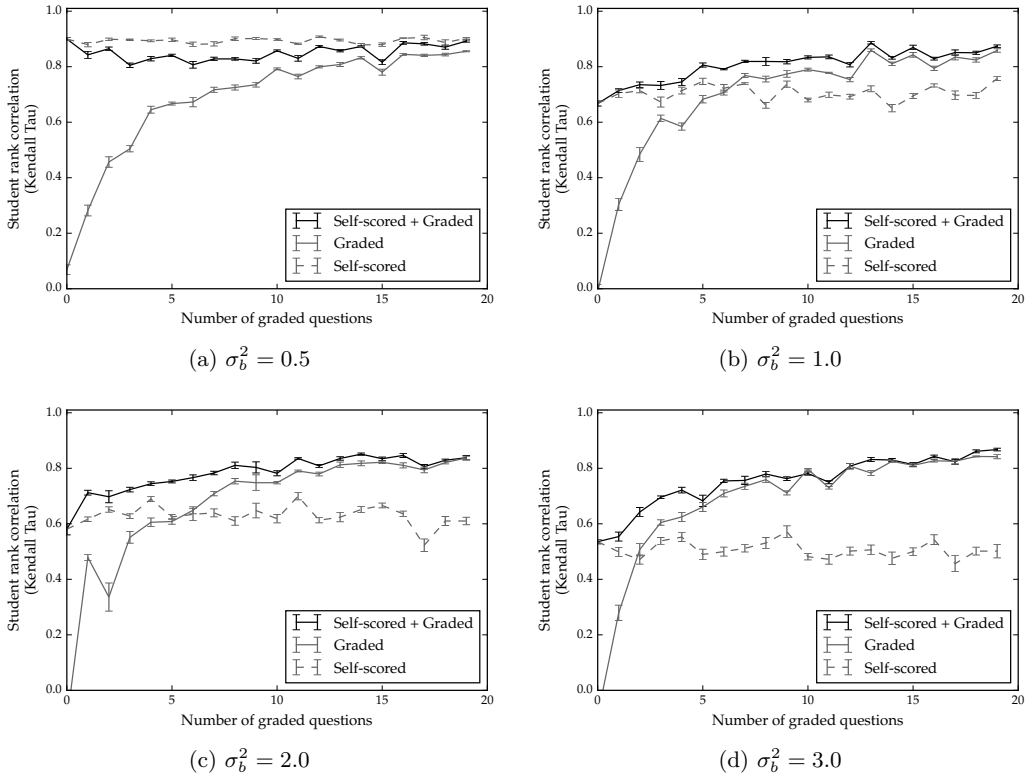
Figure 1: The optimal strategy for providing a self-assessment score $\log\theta_{ij}$ for a student with ability s_i on a question of difficulty q_j, assuming the student's knowledge that a random fraction ρ of the questions will be graded. The optimal strategy is approximately piece-wise linear as a function of the student's relative ability $s_i - q_j$. In the regime of low relative ability, the student's optimal strategy is to report a fixed score that is a function of ρ, regardless of his or her relative ability.

The optimal strategies for different values of ρ are illustrated in Figure 1. The student's knowledge of the mechanism is reflected by the appearance of a lower-bound on the self-assigned score in a region where the student is likely to do poorly (low values of $s_i - q_j$). This is expected: If the student is aware that the chance of a particular question to be graded is low enough, it would make sense to take advantage of those odds and "bet" a small, but a non-zero amount, even if the student does not know the correct answer. From a practical perspective of implementing a system that solicits self-assessment scores, it would not make sense to provide the user with the ability to provide a self-assessment score lower than their optimum. From the model inference perspective, this introduces a complication: Observations that correspond to the lowest possible self-score do not correspond to any specific $s_i - q_j$, but rather an entire range. This problem is known generally as *censored regression.* and can be solved using the same approach as for the original problem, but with the modified likelihood function that accounts for this "kink." Note that a similar restriction on the likelihood (but as an upper-bound) is introduced when the maximum attainable score for a problem is incorporated into the scoring function.

4. EXPERIMENTS

4.1 Simulations

It is insightful to study the effect of bias in the population of students on the quality of the learned parameters in the IRT model: student ability parameters and question difficulty parameters. We perform a simple simulation of a classroom with 50 questions and 30 students (question difficulties and student abilities are sampled from a zero-mean normal distribution with a standard deviation of 3), where each student answers each question (a total of 1,500 responses). In this simulation, each student submits their self-grade $\log(\theta_{ij})$ for



(a) $\sigma_b^2 = 0.5$ (b) $\sigma_b^2 = 1.0$

(c) $\sigma_b^2 = 2.0$ (d) $\sigma_b^2 = 3.0$

Figure 2: Simulation results. Rank correlation across students obtained using three models for different variance of self-grading bias (σ_2): (i) *black*: a model that uses student self-scores and the correctness of their response to a subset of graded questions (number of graded questions on x-axis), (ii) *solid gray*: a model that uses correctness of their response to a subset of graded questions only (number of graded questions on x-axis) and (iii) *dashed gray*: a model that uses only the students' self-score.

each question by optimizing their utility according to the utility function in 3. We repeat the simulation for four different populations of students, each with a different variance σ_b^2 of the bias parameter. To evaluate the quality of the inferred student parameters, we compute the rank correlation (Kendall Tau) between the true ordering of the students (by their true parameters) and the ordering obtained by sorting the students based on the inferred parameters. The Kendall Tau metric is defined as follows:

$$KendallTau(\mathbf{s}, \hat{\mathbf{s}}) = \frac{N_{\text{pairs}}^{\text{correct}} - N_{\text{pairs}}^{\text{wrong}}}{N_{\text{pairs}}}$$

where $\mathbf{s}$ and $\hat{\mathbf{s}}$ are the true and inferred student ability parameters, respectively, and $N_{\text{pairs}}^{\text{correct}}$ and $N_{\text{pairs}}^{\text{wrong}}$ is the number of student pairs that are ordered correctly in the inferred ranking (with respect to the true ranking) and the number of pairs that are ordered incorrectly, respectively. Kendall Tau is equal to $+1$ when the rankings are consistent and to -1 when the rankings are inverted. The corresponding results are shown in Figure 2.

Three models were evaluated:

- **Self-grading only**: Only students' self-submitted scores $\log(\theta_{ij})$ are used in fitting the Rasch model parameters. All students submit their self-scores for all questions. The correctness of students' responses is not used in fitting the Rasch parameters.
- **Instructor-grading only**: Only the correctness of the responses is used for fitting the Rasch model parameters; this is a classic Rasch model. We vary the number of questions used in fitting the model parameters (x-axis in Figure 2).
- **Self-grading + instructor-grading**: A combination of self-scores submitted by all students for all questions and the correctness of a subset of submitted questions is used for fitting the Rasch model parameters (number of questions used is the x-axis in Figure 2).

In the case where the students in the class are relatively unbiased (low σ_b^2) (top left in Figure 2), self-scoring achieves a better rank-correlation than the traditional IRT Rasch model, even when many questions are instructor-scored. Interestingly, in the regime of low bias, including actual instructor-graded responses actually negatively affects the correlation (this is due to over-fitting caused by a small number of instructor grades—introducing additional bias variables requires a sufficient number of observations to infer them reliably; this performance drop eventually disappears when a sufficient

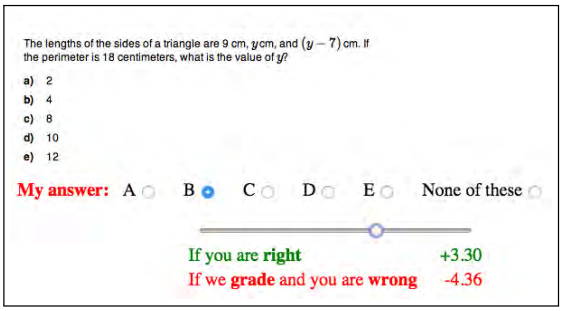


Figure 3: Screenshot of one question from the Mechanical Turk task. A subject answers a math question and provides a self-assessment score by adjusting a slider. The student sees the number of points that they will gain if they answer the question correctly (green) and the number of points they will lose if they answer the question incorrectly (red).

number of questions is included). As the bias of the population increases, the performance of the self-scoring model decreases but still exceeds the performance of the instructor-only Rasch IRT, especially in situations where only a few questions are scored.

4.2 User study

To evaluate the efficacy of the proposed self-grading approach, we conducted a user-study on Amazon's Mechanical Turk. We solicited 206 subjects to participate in a task titled "Do a short math quiz and earn bonus!". The subjects were asked to answer 30 math questions of varying difficulty levels ranging from basic arithmetic to pre-calculus. The questions from the dataset introduced by [6] were used in our experiment. All questions were multiple choice and included a "none of the above" option, included in order to minimize the probability of getting a right answer through a process of elimination. Although in practice, multiple-choice questions mostly defeat the purpose of self-grading, we use multiple choice questions for the ease of evaluation and the lack of subjectivity that would be otherwise present in free-response questions. Figure 3 illustrates a single question from the task. The subjects were asked to mark what they believed to be the correct answer, and then to assign themselves the number of points that they would receive if they answered the question correctly. The input was provided through a slider. Moving the slider automatically displayed the number of points that the subject would gain if they answered the question correctly (green), and the number of points they would lose if they answered the question incorrectly (red). The points were then converted to currency (1 point = \$0.01), and paid through a "bonus" mechanism in Mechanical Turk. We chose to use real currency as a reward to to ensure that the subjects had a stake in their performance, and thus there is incentive to think carefully about their self-assigned scores.

We follow the same evaluation scheme that we described in the previous section. Recall, that we are interested in the quality of the assessment derived from the students' self-evaluation. In the simulation study, a "gold-standard" assessment was available and allowed us to use rank correlation between the "gold-standard" ranking and the inferred ranking as an evaluation metric. In this user-study, we consider the ranking inferred by the IRT model that relies on the complete dataset, as a proxy for the "gold-standard" ranking. We then repeat the evaluation scheme described in the previous section: (i) vary the number of instructor-graded questions from 0 to all questions (30) and combine that with the self-assigned scores for every question, (ii) infer the ranking using the proposed model, and (iii) compare it to the ranking that is derived from "gold-standard" proxy.

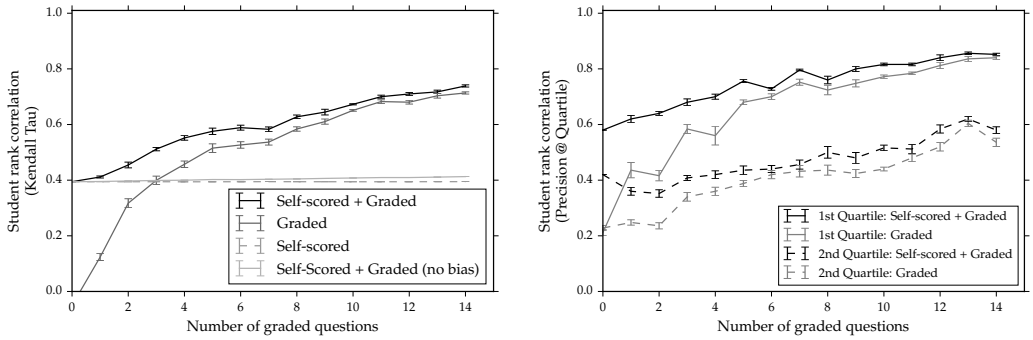
We find that the results are comparable to those obtained in the simulation (Figure 4(a)). Self-scoring is already able to obtain a reasonable correlation with the "gold-standard" ranking even without any instructor-graded question. Incorporating instructor-grades for additional questions improves the performance. Rank correlation metrics, such as Kendall Tau, while convenient for summarizing the results with a single quantity, often fail to distinguish regimes where the model might perform differently. It is instructive to consider the performance of rank-correlation in the different segments of the ranking. Figure 4(b) decomposes the results by quartiles. We employ a more intuitive metric, *Precision@Quartile*, defined as follows:

$$Precision@Q_i = \frac{|\hat{S}_{Q_i} \cap S_{Q_i}|}{|\hat{S}_{Q_i}|}$$

where S_{Q_i} is the set of students in the ith quartile of the "gold-standard" ranking, and $\hat{S}_{Q_i}$ is the set of students in the ith quartile of the inferred ranking. This metric captures the ability of the model to perform within a particular segment of the ranking. For example, looking at Precision at the first quartile, measures the ability of the model to predict top students. From Figure 4(b) we can conclude that the model is significantly better at distinguishing the top-ranked students (first quartile) as compared to the lower-ranked students (second quartile). By using the self-scoring signal without any instructor-graded questions, we are able to recover nearly 60% of the top quarter of all students. The performance in the second quartile is significantly lower, but follows the same trend: incorporating the students' self-reported scores in the regime of zero to several questions significantly improves performance over the baseline of instructor-graded questions alone. This observation leads to the conclusion that, at least in this study, better students were better at estimating their ability. We look into the effect of self-estimation performance in more detail in the next section.

4.3 Self-assessment and bias

The performance of the model that relies on self-assessment depends fundamentally on the model's estimates of the students' biases as well as the ability of the students to self-assess reliably (self-assessment variance). In our model, we infer only the individuals' biases and assume constant variance in the self-assessment likelihood (these could in principle be estimated as well. Figure 6 illustrates the individual inferred biases for each student (averaged across multiple folds), sorted in an increasing order. The resulting distribution illustrates the skew in the bias distribution towards "under-confidence," i.e., most students tend to under-estimate their ability (act conservatively). The importance of estimating bias is underlined in Figure 4(a), where we include an additional baseline **Self-Scored + Graded (no bias)** (light solid line). This baseline combines self-assessment and instructor-grades but does *not* incorporate the explicit student-bias parameter. As



(a) (b)

Figure 4: User study results. Rank correlation across students obtained using three different models (i) **Self-scored**: a model that relies entirely on student-submitted self-assessments, (ii) **Graded**: a model that relies entirely on instructor-provided grades, as a function of the number of graded questions (x-axis), and (iii) **Self-scored + Graded**: a model that aggregates students' self-assessment scores on all questions and a variable number of instructor-graded questions (x-axis). (a) Computes rank correlation across all students using Kendall Tau, and (b) decomposes rank correlation across the first two quartiles using the *Precision@Quartile* metric. The model that combines self- and instructor-assigned scores is significantly better at predicting the top-performing students (first quartile). Combining instructor grades with self-assessment significantly improves both rank measures, especially when only a few questions are graded. Note that the total number of questions in the study was 30; we display the results up to 15, as the differences between both models is not substantial beyond that.

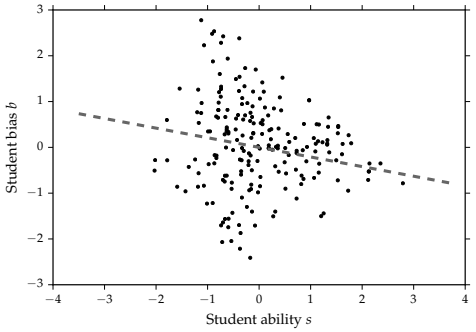


Figure 5: Bias vs. ability (centered). Both parameters were inferred using all of the available data. Each point in the scatter-plot corresponds to one student. A weak, but significant correlation between bias and ability exists.

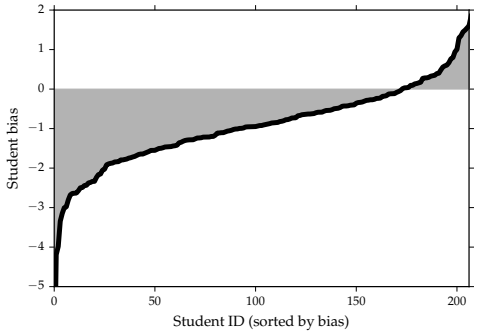


Figure 6: Inferred bias parameter of each student (sorted in an increasing order). The bias parameter was inferred using all of the available data.

evident from the graph, estimating bias is critical for combining self-grading and instructor-grading: without the bias parameter, the model is not able to leverage the benefits of both signals.

It is potentially insightful to investigate the relationship between self-assessment bias and ability. We consider the inferred bias parameter after incorporating instructor-grades for all questions, and compare it to the inferred ability parameter of each student. The result is illustrated in the scatter-plot in Figure 5. While the relationship between the two is not strong, there exists a negative correlation between ability and self-assessment bias (Pearson's correlation: 0.17, p-value = 0.013). Students that are more able tend to underestimate their ability, and students that are less able tend to inflate their ability. This finding is consistent with the literature in self-assessment [17, 16].

5. CONCLUSION AND FUTURE WORK

In this work, we have developed a novel approach for performing calibrated, summative self-assessment by combining (i) student's self-evaluations obtained via an incentive-compatible scoring mechanism and (ii) a minimal number of instructor-graded responses. We have shown that when the scoring rule is quadratic, the standard IRT Rasch model reduces to standard linear regression. We have demonstrated that the quality of the inferred assessment using self-scoring alone without additional instructor input is, on-average, comparable to the performance obtained using the standard IRT that requires significant instructor effort. Furthermore, by incorporating a minimum number of instructor-graded responses, we have shown that our approach substantially improves the estimates of the students' abilities and the

questions' difficulties. Finally, we have addressed the long-standing issue of applying scoring rules in practice: dealing with the consequences of individuals' biases and non-risk-neutrality. We have proposed to explicitly model the combined effect of these two factors within the standard IRT framework, allowing the model to effectively de-bias these individual differences.

Our results open an interesting direction of inquiry: are there other scoring functions that are more efficient at estimating IRT parameters, and if so, can the scoring functions be adapted to individual students and questions, improving the efficiency of adaptive testing? In order to facilitate further research in this direction, we release all code and data used in this study.

6. ACKNOWLEDGMENTS

We would like to thank Mr. Lan, Ph.D., as well as A. E. Waters and R. G. Baraniuk for their help with the Mechanical Turk dataset [6]. The work of I. Labutov was supported in part by a grant from the John Templeton Foundation provided through the Metaknowledge Network at the University of Chicago. The work of C. Studer was supported in part by Xilinx Inc. and by the US NSF under grants ECCS-1408006 and CCF-1535897.

7. REFERENCES

[1] J. E. Bickel. Scoring rules and decision analysis education. *Decision Analysis*, 7(4):346–357, 2010.

[2] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

[3] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[4] G. W. Harrison, J. Martínez-Correa, and J. T. Swarthout. Inducing risk neutral preferences with binary lotteries: A reconsideration. *Journal of Economic Behavior & Organization*, 94:145–159, 2013.

[5] C. Kulkarni, K. P. Wei, H. Le, D. Chia, K. Papadopoulos, J. Cheng, D. Koller, and S. R. Klemmer. Peer and self assessment in massive online classes. In *Design Thinking Research*, pages 131–168. Springer, 2015.

[6] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *The Journal of Machine Learning Research*, 15(1):1959–2008, 2014.

[7] A. H. Murphy and R. L. Winkler. Scoring rules in probability assessment and evaluation. *Acta psychologica*, 34:273–286, 1970.

[8] J. Park and K. Williams. The effects of peer-and self-assessment on the assessors. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, pages 249–254. ACM, 2016.

[9] A. Peysakhovich and M. Plagborg-Møller. Proper scoring rules and risk aversion. *Available at SSRN 2019078*, 2012.

[10] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller. Tuned models of peer assessment in moocs. *arXiv preprint arXiv:1307.2579*, 2013.

[11] K. Raman and T. Joachims. Methods for ordinal peer grading. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1037–1046. ACM, 2014.

[12] P. M. Sadler and E. Good. The impact of self-and peer-grading on student learning. *Educational assessment*, 11(1):1–31, 2006.

[13] L. J. Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66(336):783–801, 1971.

[14] R. Selten. Axiomatic characterization of the quadratic scoring rule. *Experimental Economics*, 1(1):43–62, 1998.

[15] R. Selten, A. Sadrieh, and K. Abbink. Money does not induce risk neutral behavior, but binary lotteries do even worse. *Theory and Decision*, 46(3):213–252, 1999.

[16] B. Strong, M. Davis, and V. Hawks. Self-grading in large general education classes: A case study. *College Teaching*, 52(2):52–57, 2004.

[17] K. Topping. Self and peer assessment in school and university: Reliability, validity and utility. In *Optimising new modes of assessment: In search of qualities and standards*, pages 55–87. Springer, 2003.

[18] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.

MOOC Learner Behaviors by Country and Culture; an Exploratory Analysis

Zhongxiu Liu
North Carolina State University
Raleigh, NC
zliu24@ncsu.edu

Rebecca Brown
North Carolina State University
Raleigh, NC
rabrown7@ncsu.edu

Collin F. Lynch
North Carolina State University
Raleigh, NC
cflynch@ncsu.edu

Tiffany Barnes
North Carolina State University
Raleigh, NC
tmbarnes@ncsu.edu

Ryan Baker
Teachers College, Columbia University
New York, NY
ryanshaunbaker@gmail.com

Yoav Bergner
Educational Testing Service
Princeton, NJ
ybergner@gmail.com

Danielle McNamara
Arizona State University
Phoenix, AZ
dsmcnamara1@gmail.com

ABSTRACT

The advent of Massive Online Open Courses (MOOCs) has led to the availability of large educational datasets collected from diverse international audiences. Little work has been done on the impact of cultural and geographic factors on student performance in MOOCs. In this paper, we analyze national and cultural differences in students' performance in a large-scale MOOC. We situate our analysis in the context of existing theoretical frameworks for cultural analysis. We focus on three dimensions of learner behavior: course activity profiles; quiz activity profiles; and most connected forum peer or *best friends*. We conclude that countries or associated cultural clusters are associated with differences in all three dimensions. These findings stress the need for more research on the internationalization in online education and greater intercultural awareness among MOOC designers.

1. INTRODUCTION

Over the past decade there has been a substantial increase in the study of cross-cultural behaviors in e-learning systems. Prior researchers have shown that learners from different cultures behave differently when using educational systems, particularly in terms of their off-task behaviors [25, 21], help-seeking [21], and collaboration [22, 16]. The cultural differences uncovered in these studies suggest that designers of future e-learning platforms would benefit from a better understanding of their distinct target populations and distinct cultures.

Large-scale MOOCs typically attract diverse international audiences. The course we discuss here, for example, attracted students from 172 countries on 5 continents. Despite this acknowledged diversity, most MOOCs take a one-size-fits-all approach to designing and structuring the course. The materials are typically offered in a single format and language, or via direct translations that preserve the structure, pacing, and content.

Prior researchers have shown that country of origin affects students' performance in MOOCs. Nesterko et al. [19] found that non-American students were more prone to complete MOOCs and to seek certification than their U.S. counterparts. Guo and Reinecke [12] found that a student's country of origin significantly predicted the amount of content that they would cover and the amount of time that they spent reviewing prior course content. Kizilcec [17] found that there was a significant correlation between a country's level on the Human Development Index and the number of students from that country who completed a majority of the assignments. In each of these studies, however, nationality was treated as a single independent factor. No substantive comparisons were made between countries or cultures, nor did the authors frame their conclusions in the context of prior theoretical work on cultural differences in learning.

A deeper understanding of how students differ both within and across cultures will help us to design and deploy more effective, and truly international MOOCs. And this understanding will be enriched by relating these differences to the rich existing literature on cross-cultural education such as Hofstede's cultural dimensions theory [13] and the Cultural Dimensions of Learning Framework (CDLF). In this paper we will address this need through our analysis of cross-cultural student behaviors in an existing MOOC. This was an open course with a total enrollment of 29,149 students drawn from 172 countries and 5 continents. We found clear inter-country and inter-cultural differences in the observed

student behaviors and in the distribution of user categories. We also found that these differences can be evaluated in the context of existing theoretical frameworks and that they are consistent with prior educational literature.

2. LITERATURE REVIEW

2.1 Culture & Educational Technology

Advances in educational technology have enabled educators to incorporate technologies at larger scale and to collect richer and more diverse educational data than ever before. This has, in turn, substantially increased interest in studying variations in the use of e-learning tools across cultures.

One approach to understanding the impact of culture on learning is through field observation. Rodrigo et al. [25] coded U.S. and Filipino students' on- and off-task behaviors when using three ITSs. They found that Filipino students spent more time on task than their U.S. counterparts on all three systems. They also found that the Filipino students gamed some systems more than others. Similarly, Ogan et al.[22] coded the on-task behaviors and interaction of similar students in Chile. They found that the Chilean students had a higher proportion of on-task interactions than the U.S. students studied previously.

Another approach is through educational data mining. Ogan et al. [21] generated student models from ITS logs collected in three countries: Costa Rica, The Philippines, and the U.S. Their goal in this work was to predict effective help-seeking behaviors. They found that it was possible to generalize the U.S. model to Filipino students but not to students from Costa Rica. Saarela and Karkkainen [27] applied a hierarchical clustering algorithm to data collected from the PISA, a worldwide assessment of 15-year old students covering reading, mathematics, and science. They found that students' performance on the test clustered by country, suggesting cultural influences.

While these studies found interesting cross-cultural differences, we have little understanding of why these differences occur, or of how they relate to more general cross-cultural variation. Learning behaviors are influenced by a complex set of factors and cross-cultural comparisons may help us deepen our understanding of this phenomenon and highlight ways to remediate or accommodate it. In this paper, we explore the logs of student activity in a MOOC, with an eye toward how culture may relate to differences in behavior.

2.2 MOOC Research

MOOCs represent both opportunities and challenges for educators. On the one hand they involve large numbers of users working in highly instrumented systems which can, in turn, provide deep insights. On the other hand, however, MOOCs have high dropout rates, wide variation in levels of engagement, and MOOC users have extremely diverse motivations and demographic backgrounds. Thus any insights are qualified by the noisy nature of the data. Researchers have therefore focused their efforts on better understanding MOOC users and their differing behavior patterns.

One approach to understanding MOOC students is to build predictive behavior models based upon their clickstream data, such as mining sequences of actions for analysis [29, 5]. These induced models are highly accurate but are not always readily interpretable. Other work has focused on improving our understanding of engagement and dropouts by detecting key subgroups. In this work, researchers have used hierarchical clustering to identify groups of students with similar patterns of engagement, such as those who viewed many lectures but rarely attempted quizzes, and those who balanced their activities equally [17, 10, 4, 1]. Kizilcec et al. [17] and Ferguson et al. [10], for example, clustered students by engagement factors such as the number of lectures viewed and quizzes attempted. Anderson et al. [1] likewise used lecture views and considered the ratio of lectures to assignments while Bergner et al. [4] focuses solely on assignments attempted. These studies served to highlight the distinct behavioral patterns of different subgroups.

Researchers have also begun to study students' diverse backgrounds through voluntary surveys with the goal of understanding how their incoming motivation [28, 2] and demographic features [19, 17, 12] affect their observed behaviors. Both Nesterko [19] and Deboer [9] found that participation (as indicated by survey responses) and certificate attainment rates differed across countries, continents, and genders; they did not, however, delve deeper into students' in-system behaviors as logged by the learning environment. Wang and Baker [28], by contrast, found that learners receiving course certificates tended to be more interested in course content, while students not receiving certificates often stated that they were seeking a new type of learning experience.

Few of these researchers however, have focused on the relationship between geographic information and observed behaviors. Guo and Reinecke [12] applied linear regression to correlate some demographic features such as years of education to geographic data. They found that a students' country of origin was significantly related to their coverage of the course content overall and the extent to which they reviewed prior content, called *backjumps*. They attributed this diversity to varying student-to-teacher ratios. They found that countries with a higher ratio had a higher frequency of backjumps suggesting more time on review. In related work Kizilcec focused on partitioning countries into tiers based upon the Human Development Index (HDI). They found that as the HDI tier increased, so to did the proportion of students who completed the course. While these results are instructive, however, the authors made no attempt to situate these results in the context of existing theoretical models of cross-cultural learning.

Thus the results from prior MOOC research show that understanding students' diverse backgrounds can be essential to the development of effective educational interventions, and to providing useful support for student engagement and participation. Geographical location, considered as a set of economic, cultural, and educational differences, may play a crucial role in understanding, supporting, and appealing to the increasing population of MOOC users.

2.3 Theoretical Frameworks

MOOCs and educational technologies allow us to collect robust information about cross-cultural differences in user behaviors. Yet we face challenges in interpreting and explaining these results in a consistent theoretical framework.

Prior educational researchers have worked to identify related cultural dimensions and values, and to examine how they vary across cultures. One common framework is Hofstede's Cultural Dimensions Theory [13, 14]. Hofstede analyzed a set of 117,000 attitude surveys collected by IBM from their international workforce and synthesized a set of 7 general cultural dimensions: a) power distance; b) collectivism vs. individualism; c) femininity vs. masculinity; d) uncertainty avoidance; e) long/short term orientation; and f) indulgence vs. restraint. Hofstede then calculated scores for each culture within these dimensions.

Hofstede's dimensions have been used to analyze and explain differences in collaboration across cultures [16], as well as differences in help-seeking and off-task behavior in educational technology [21, 25]. However these studies have suggested that the cultural dimensions framework has some limitations in explaining these findings. Many of the key differences in the observed behaviors do not correspond to the differences that Hofstede's theory suggests. In particular, variations in collectivism and collaboration/help-seeking strategies do not seem to relate well to Hofstede's underlying dimensions. Therefore we will combine this with the Cultural Dimensions Learning Framework (CDLF).

The CDLF framework, designed by Parrish et al. in 2010 [24], defines eight cultural parameters regarding social relationships, epistemological beliefs, and temporal perceptions, and how they manifest in learning situations. The CDLF has been used to guide the design and analysis of e-learning across cultures [23, 15]. For the purposes of our analysis we will focus on the intersection of the CDLF and the Hofstede dimensions. We will use this hybrid framework to group countries into cultural clusters, and to interpret the observed behavioral differences between them. Table 3 provides an overview of the shared dimensions.

While these frameworks may help to explain observed behaviors, it is worth noting that learner behaviors in MOOCs can be affected by many other factors such as personal motivation. Wang and Baker [28], for example, surveyed the motivations of incoming students on a later version of the course we study here and found that learners who obtained course certificates tended to be more interested in course content than those who took the MOOC in order to test the learning experience. While this highlights the importance of individual differences, our analysis below we will focus on inter-country differences and cultural factors.

Table 1: Intersection of Hofstede Dimensions and the Cultural Dimensions of Learning Framework.

Hofstede Dimension [13]	**Selected Interpretations in CDLF [24]**
Power Distance: the extent to which the less powerful members expect and accept unequal/unfair situations	Countries with high power distance view teacher as an unchallenged authority and the primary communicator, not as a fallible peer.
Individualism: the degree of interdependence a society maintains among its members	Highly individualist students are more prone to speak up in class, to value diverse opinions in learning, and to be motivated by personal gain.
Masculinity: the degree to which a culture is motivated by competition (instead of life quality)	More masculine cultures are associated with increased levels of competition and a heavier pursuit of recognition.
Uncertainty Avoidance: The extent to which a culture feels threatened by ambiguous or unknown situations and tries to avoid these	Students who avoid uncertainty tend to focus more on getting the right answer from authoritative sources and from the structured learning activities.

3. DATA

The data used in this study was collected from Big Data in Education (BDE), an 8-week long MOOC offered by the Teacher's College at Columbia University on the Coursera platform [28]. The BDE curriculum included video lectures, discussion forums, and 8 weekly assignments or quizzes. The lectures covered key methods for educational data analysis. The assignments required students to analyze existing data (typically real data collected from educational settings) and to answer questions about their results. All of the assignments were automatically graded via numeric or multiple-choice questions. Students were given between 3 and 5 attempts to complete each assignment with the best score being counted. Students were required to complete their assignments within 2 weeks of it being released. In order to obtain a certificate students were required to obtain an average grade of $\geq 70\%$ over all 8 assignments. High performing students could receive a certificate with distinction. 638 students completed the course and obtained a certificate.

Data from this course has been previously used to study motivation [28], negativity [7], student communities [6], the relationship between linguistic quality of forum posts and completion[8], as well as longitudinal behavior patterns[31].

For the purposes of our analysis we analyzed clickstream data containing user IDs, IP addresses, URLs and timestamps for 29,149 students. This data included all 638 students who received a certificate as well as 750 who posted on the forum. After classifying students by behavior type we found that a total of 1,591 students were actively engaged with the course while the remaining 27,588 were 'bystanders' who enrolled but did not do any significant work. We assigned users to regions based upon their most frequent IP address as has been done in prior work [17, 9, 12]. The top 15 countries by registration are shown in Figure 1.

We then analyzed the URLs located in the clickstream data to identify the following major activities: view lecture (VL), attempt or submit quiz (AQ, SQ), and read or make a post in forum (RP, MP). We then generated activity sequences from this data using an n-gram approach consistent with

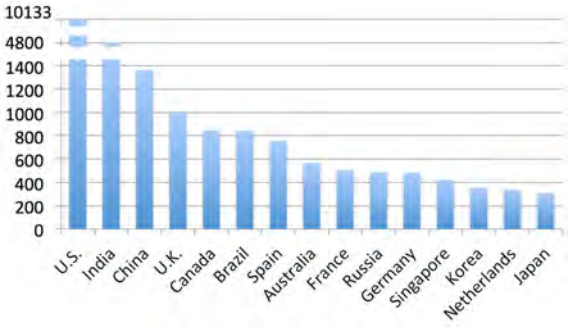


Figure 1: Number of Registrants from the Top 15 Countries with Most Registrants

prior research [29, 5]. Note that this data does not contain information about how long the student spent viewing a URL. The data only records individual mouseclicks. Therefore it functions as a record of student access but not a reliable indicator of engagement.

4. METHODS AND RESULTS

We hypothesize that students from different countries or cultures will behave differently in the course. We chose to examine four research questions: *RQ1. (Course Activity Profiles, CAPs)* What are the primary categories of students based upon the frequency (both total and relative) with which they accessed different course activities? *RQ2. (CAPs by Country)* Does the proportion of student categories differ by country? *RQ3. (Quiz Activity Profiles, QAPs)* When do students in each category access the different types of course activities and how is that correlated with quiz submissions? *RQ4. (QAPs by Culture & Country)* How do quiz-based activity profiles and countries relate to the four overlapping Hofstede/CDLF cultural dimensions of: power distance, individualism, masculinity, and uncertainty avoidance? *RQ5. (Forum best friends)* Is a student's most frequent forum partner in the same country/culture?

For RQ1, we used hierarchical clustering to identify five course activity profiles (CAPs) (e.g. students who focused solely on quizzes). For RQ2, we clustered countries by the proportion of students who fit each CAP in order to determine whether or not students from a given country are more likely to fit one CAP over another. For RQ3, we partitioned the course data by quizzes and examined whether or when students in each CAP accessed the lectures, quizzes, and forum content. This led to the development of Quiz Activity Profiles (QAPs). For RQ4, we then clustered students based upon their cultural dimensions and compared the QAPs by culture and student category (CAP). For RQ5, we performed a χ^2 analysis to investigate whether the students' most frequent interlocutor on the forums were more likely to be drawn from the same country/culture. In each section below, we will present the methods and results for each of these questions in greater detail.

4.1 RQ1: Course Activity Profiles, CAPs

What are the primary categories of students based upon the frequency (both total and relative) with which they accessed different course activities? Prior researchers have used hierarchical clustering to discover meaningful subgroups such as: users who viewed many lectures but rarely attempted quizzes and users who balanced the number of lectures viewed and quizzes attempted [17, 10, 4, 1].

In this work we applied hierarchical clustering to classify students based upon the proportion of activities that they engaged in over the course. These included: lectures accessed, quizzes attempted, and form posts made or accessed. We found that clustering students by the the number of lectures that they accessed and quizzes attempted yielded five interpretable clusters which we designated *solvers* (generally take more quizzes), *viewers* (generally watch more lectures), *all-rounders* (balance both), *samplers* (watch some lectures and do a quiz), and *bystanders* (do very little). Table 2 shows the CAP clusters with average silhouette widths (ASWs) in excess of 0.68, which indicates that they are well-chosen classifications [26]. These CAPs closely resemble the student types described by Anderson et al. [1] who clustered MOOC students based upon the ratio of lectures viewed to assignments completed. In this case we used attempts in place of submissions.

Table 2: Course Activity Profile Clusters: size, #lectures viewed, #quiz attempts, and performance.

			% Certificate	
CAP	**Lectures viewed (max:54)**	**Quiz Attempts (max:7)**	Distinct	Normal
Solver (n=388, ASW=0.72): mainly attempt quizzes	M:5.30 Sd:7.15	M:7.67 Sd:0.77	41.10%	0.07%
Viewer (n=107, ASW=0.72):mainly view lectures	M:49.57 SD:2.95	M:0.55 SD:0.96	0%	0%
All-rounder (n=519, ASW=0.68):balance lectures & quizzes	M:45.23 Sd:8.3	M:7.58 Sd:0.89	79.19%	8.29%
Bystander (n=27558, ASW=0.84):do little	M:1.87 Sd:2.72	M:1.25 Sd:1.43	0%	0%

As Table 2 shows, the all-rounders have the highest rate of certificate completion. For the rest of our analysis we will focus on three categories: viewer, solver, and all-rounder.

4.2 RQ2: CAPs by Country

Does the proportion of student categories differ by country?

After identifying the meaningful CAP clusters, we compared countries based upon the proportion of CAPs observed. We again applied hierarchical clustering on countries with more than 15 users from the viewer, solver, and all-rounder students. In this case we found that three clusters yielded the highest ASW values. These clusters are shown in Figure 2.

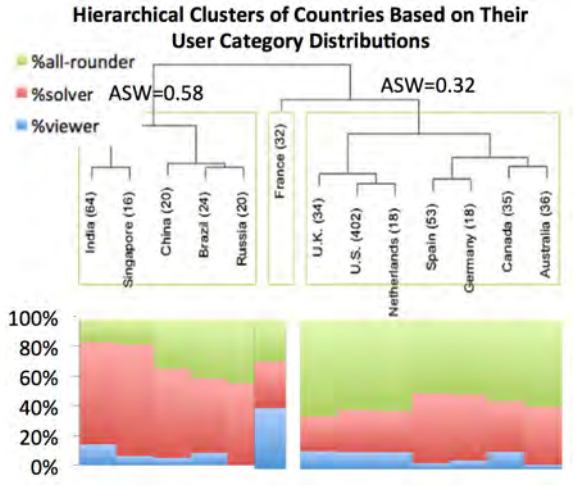


Figure 2: Hierarchical clusters of countries by proportion of user categories. For each country, the proportion of user categories is plotted as stacked bar, and the sample size is given in parentheses

This clustering grouped countries with a high proportion of solvers in Cluster 1. This includes developing countries, Russia, and Singapore. The proportion of solvers present in Cluster 1 is significantly higher than that of cluster 3: $\chi^2(1, N = 740) = 34.95, p < 0.001$.

4.3 RQ3. Quiz Activity Profiles, QAPs

When do students in each category access the different types of course activities and how is that correlated with quiz submissions?

After identifying the CAPs and examining their relative proportion within countries, we proceeded to analyze the inter-country behavioral differences within each CAP. It is our hypothesis that students from different countries will behave differently given the different Hofstede/CDLF dimensions. In order to assess this hypothesis we analyzed the behavioral differences among users with regards to the course content accessed in the three learning phases described below.

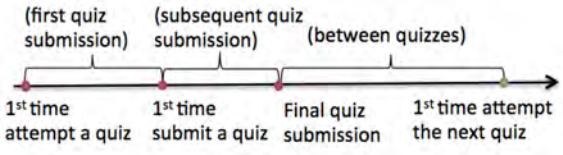


Figure 3: Illustration of the three learning phases

In order to better understand when students engaged in different learning activities we segmented the activity sequences into three phases based upon the quiz attempts. These phases are shown in Figure 3. For each phase we counted average number of lectures viewed (VL), forum posts made (MP), and posts read (RP). For the first quiz submission, and for the subsequent submission phases, we also counted the average number of times that a student attempted and submitted the same quiz (AQ, SQ). We excluded viewers from this analysis as they made little to no attempts at the quizzes.

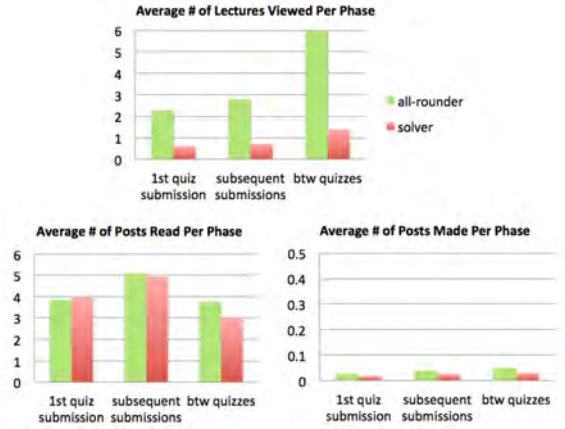


Figure 4: Quiz Activity Profiles for Solvers and All-rounders in Three Learning Phases.

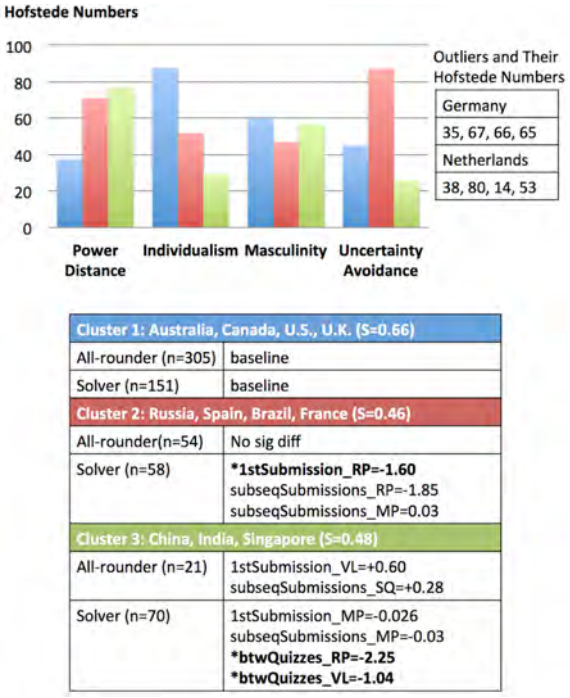
The relative QAP values for solvers and all-rounders in this analysis are shown in Figure 4. We then conducted a series of pairwise Kruskal-Wallis tests [20] with Benjamini-Hochberg correction [3] comparing the performance by group and learning phase to a baseline of the course average. We found that the solvers and all-rounders viewed significantly more lectures between the quizzes and read more posts during subsequent quiz submissions than in the other learning phases.

4.4 RQ4. QAPs by Culture

How do quiz-based activity profiles and countries relate to the four overlapping Hofstede/CDLF cultural dimensions of: power distance, individualism, masculinity, and uncertainty avoidance?

In order to assess this question we applied hierarchical clustering on countries with more than 15 all-rounders, solvers or viewers, based on the four shared dimensions. This produced three clusters with ASWs above 0.46. For our analysis we treated the first cluster as the baseline as it contains the majority of the student population. Then, for each course activity in the learning phases, we conducted a series of Kruskal-Wallis tests comparing each QAP by CAP and Cluster with the course average baseline. We applied Benjamini-Hochberg correction to correct for the multiple tests as above. The results are shown in Figure 5.

Countries in cultural cluster 1 (Australia, Canada, the U.S. and U.K. cluster) have the lowest average power distance and the highest average individualism. In our analysis we found that solvers in clusters 2 (Russia, Spain, Brazil, & France) and 3 (China, India, & Singapore) read and made more posts during multiple learning phases. These differences were significant or marginally-significant. Moreover, solvers in cluster 3, whose countries are characterized by the highest average power distance and lowest average individualism, viewed significantly fewer lectures between the quizzes. All-rounders in cluster 3 also viewed significantly fewer lectures during the first quiz submission and made



Cluster 1: Australia, Canada, U.S., U.K. (S=0.66)	
All-rounder (n=305)	baseline
Solver (n=151)	baseline
Cluster 2: Russia, Spain, Brazil, France (S=0.46)	
All-rounder(n=54)	No sig diff
Solver (n=58)	***1stSubmission_RP=-1.60** subseqSubmissions_RP=-1.85 subseqSubmissions_MP=0.03
Cluster 3: China, India, Singapore (S=0.48)	
All-rounder (n=21)	1stSubmission_VL=+0.60 subseqSubmissions_SQ=+0.28
Solver (n=70)	1stSubmission_MP=-0.026 subseqSubmissions_MP=-0.03 ***btwQuizzes_RP=-2.25** ***btwQuizzes_VL=-1.04**

Figure 5: Cultural clusters based on Hofstede-CDLF values with statistically-significant values ($p \leq 0.05$ bolded) and marginally significant ($p \leq 0.1$) QAP differences as compared to baseline behaviors.

more submissions per quiz, this difference was marginally significant.

We found a high degree of overlap between the cultural clusters and the CAP clusters described in section 4.2. Cultural cluster 1 is a subset of the all-rounder CAP cluster, by country, and cultural cluster 3 is a subset of the solver CAP cluster. These results suggest that students from countries with higher individualism and lower power distance are twice as likely to be all-rounders, while students from countries with higher power distance and lower individualism are more prone to focus on evaluations. However, cultural cluster 2 includes students that were evenly split between solvers and all-rounders. These findings suggest that the cultural dimensions are directly connected to some aspects of the students' observed behaviors, but other personal motivations may also dominate student behaviors.

4.5 RQ5. Forum "Best Friend"

Is a student's most frequent forum partner in the same country/culture?

For this analysis we identified each students' "best friend" based upon their forum interactions. In a prior study, we tested whether we can predict students' performance in the course based upon their implicit social relationships in the forum [6]. In this case we constructed a similar relationship graph for the 750 forum users based upon that work and the work of Fire et al. [11]. Edges in the graph were weighted based upon the number of times that a user had replied to a thread that the other used had posted in. We then defined a students' "best friend" as the individual with the highest-weighted edge between them.

Then, for each of the top 15 countries and the 3 cultural clusters defined in the prior section we performed a χ^2 test with the proportion of "best friends" within the cluster as the dependent variable. Our goal was to test whether or not the cluster was a significant predictor of the proportion of individuals with "best friends" in their cluster. The results are shown in Table 3. We found that for all three cultural clusters, the students are significantly more likely to have a best friend within their own country.

Table 3: Groups whose "best friends" are significantly more likely to be from the same group

Clusters & Countries	% IN this group with best friends in this group	% NOT IN this group with best friends in this group	p
Cluster1(n=381): Australia, Canada,U.S.,U.K.	64.04%	54.09%	0.0065
Cluster2(n=83): Russia, Spain, Brazil, France	36.60%	5.93%	<0.001
Cluster3(n=91): China, India, Singapore	19.78%	10.13%	0.0066
China (n=19)	26.32%	1.99%	<0.001
Brazil (n=38)	63.16%	1.31%	<0.001

5. DISCUSSION

In this study, we conducted an exploratory analysis on three dimensions of MOOC behavior by country and culture. We first identified five Course Activity Profiles (CAPs) based on the number of lecture views and quiz attempts: viewers, solvers, all-rounders, samplers, and bystanders. We found that the all-rounder students were most likely to obtain a certificate of completion, followed by the solvers. This indicates that the behavior profiles exhibited by these groups are a good indicator of students who are working toward certification.

We then studied the distribution of CAPs over countries. To that end we clustered countries with 15 or more students in the solver, viewer, or all-rounder categories based upon their CAP distributions. Interestingly we found that the developing countries in our dataset all contained a substantially higher proportion of solvers than other countries. We then clustered the same set of countries using the Hofstede/CDLF cultural frameworks [13, 24]. We found that the resulting cultural clusters also aligned with the observed student types. Our first cultural cluster, which included Australia, Canada, the U.S., and the U.K., was dominated by all-rounders while our third cluster, which included China, India, and Singapore, was dominated by solvers. This distinction may reflect differing educational traditions, as Asian countries are historically more test-centric [18, 30]. It may also reflect differences in the professional environments of the countries as certificates may be more valuable for ca-

reer advancement in Asian nations. Indeed, it may be the case that the solvers are studying offline and are using the MOOC as a certification system.

Following that we focused on the students' quiz-centric behavior. We defined the Quiz Activity Profiles (QAPs) based upon the students' major activities between quizzes and before subsequent quiz attempts. We found that, regardless of the student's CAP, they typically viewed lectures between quizzes, and then turned to forum posts after their initial submission and before any resubmission. This resembles some traditional classroom settings where students attend lectures before doing homework and then only turn to the office hours or peers after they face some difficulty.

When clustering the countries by cultural dimensions we also found that two of our clusters were dominated by countries with higher power distances and lower individualism (cluster 2: Russia, Spain, Brazil, and France; cluster 3: China, India, & Singapore). Students in these clusters were less likely to interact on the forum in most of the learning phases than the students in cluster 1 which was dominated by countries with low power distance and high individualism. This finding is consistent with other work on the CDLF which found that students in countries with high power distance tend to treat the teacher as the unchallenged communicator versus students in countries with low power distance who place a higher value on dialogue and discussion in the learning process. This framework, however, does not explain the other observed variations in cultural cluster 3, notably their apparent focus on work between quiz attempts. We believe that the explanation may lie in the educational culture of this cluster. As noted above this cluster consists entirely of Asian nations which are historically test-driven. We believe that this educational culture may cause the students to view quizzes as the primary goal, leading them to focus their efforts on viewing lectures and forums after they have seen the quiz. Moreover, this cultural emphasis on exams may be the primary reason that Asian students were more prone to re-submit quizzes rather than moving on to new material.

Finally, we analyzed students' "best friends" on the forums. We found that students are more likely to have a "best friend" [6, 11] from countries in the same cultural cluster as their own. Chinese and Brazilian students, in particular, are more likely to have "best friend" from their own country. This close connection may be explained by several factors. First, students from the same country may have the same motivations and overall view of the course which would lead them to join forums that fit their shared needs. Second, students may face difficulties in communicating with individuals from other nations due to language barriers, thus making them more connected to their neighbors. And third, the observed relationships in the forums may reflect real offline relationships among students who joined the class together and are collaborating offline. In the absence of additional data we cannot distinguish among these alternatives.

Ultimately we conclude that students from different countries and cultures do exhibit different learner behaviors on the BDE MOOC. These differences may be explained by country, cultural dimensions, and educational differences. We believe that the students' observed behaviors are driven in part by their own goals and their unique cultural background. Students who come from countries that value discussion are more prone to interact on the forums. Students who come from countries that are test-centric are more prone to focus on improving their quiz scores and will structure their efforts around that. These findings contribute to our understanding of the role that culture and country play in MOOC learner behaviors. They also suggest some culturally-influenced behaviors that MOOC designers should consider when designing their materials.

5.1 Conclusions & Future Work

Our goal in this study was to increase general understanding of behavioral differences in MOOC populations, and the possible role that country and culture may play. We found interpretable inter-country and intercultural differences in students' observed activities, both across the whole course and when segmented by quizzes. We also found that forum users were most strongly connected to individuals from their own country or from culturally-related countries. We analyzed these findings in the context of a hybrid Hofstede/CDLF cultural framework and found that our observed clusters were consistent with the theoretical literature.

This paper is one of the first to explore the relationship between observed behaviors and learners' country or culture. In future work we plan to examine the generality of these findings by analyzing other related MOOCs. Our present dataset includes 29,149 accounts identified from the clickstream data, only 1,591 of which were non-bystanders, and only 750 of whom participated in the forum. While this is consistent with other MOOCs, it is also somewhat skewed and contains relatively small samples for many countries.

As we build a better understanding of the interactions between culture, behavior, and MOOC performance, new questions arise for MOOC designers. Should e-learning platform designers intervene to change cultural behaviors? For example, should they encourage students to use forums more or to communicate across cultural lines? Or should they consider supporting many separate groups by providing language-specific forums and tailored tracks? If so, how can we assess the impact of such interventions? It may be worthwhile to conduct more user-centered research so that we can better understand the unique needs of diverse populations. This type of work may help us to better understand *how* to address the diverse needs of such unprecedented student populations.

6. ACKNOWLEDGMENTS

The author would like to thank Abhishek Agrawal and Dhyey Shah for segmenting clickstreams and identifying students' geographical locations in this MOOC. This work was partially supported by NSF grant no. 1418269

7. REFERENCES

[1] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Engaging with massive online courses. In *23rd ACM Int. conf. on World Wide Web*, pages 687–698, 2015.

[2] M. Barak, A. Watted, and H. Haick. Motivation to learn in massive open online courses: Examining

aspects of language and social engagement. *Computers & Education*, 94(49-60), 2016.

[3] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 289-300, 1995.

[4] Y. Bergner, D. Kerr, and D. E. Pritchard. Methodological challenges in the analysis of MOOC data for exploring the relationship between discussion forum views and learning outcomes. In *The 8th Int. conf. on Educational Data Mining*, 2015.

[5] C. Brooks, C. Thompson, and S. Teasley. A time series interaction analysis method for building predictive models of learners using log data. In *In the Fifth Int. conf. on Learning Analytics And Knowledge*, pages 126–135, 2015.

[6] R. Brown, C. Lynch, Y. Wang, M. Eagle, J. Albert, T. Barnes, R. Baker, Y. Bergner, and D. McNamara. Communities of performance & communities of preference. In *the 2nd Int. Workshop on Graph-Based Educational Data Mining.*, 2015.

[7] D. Comer, R. Baker, and Y. Wang. Negativity in massive online open courses: Impacts on learning and teaching and how instructional teams may be able to address it. *Journal of the Center for Excellence in Teaching and Learning*, 10:92–106, 2015.

[8] S. Crossley, D. McNamara, R. Baker, Y. Wang, L. Paquette, T. Barnes, and Y. Bergner. Language to completion: Success in an educational data mining massive open online class. In *The 8th Int. conf. on Educational Data Mining*, 2015.

[9] J. DeBoer, G. S. Stump, D. Seaton, and L. Breslow. Diversity in mooc students' backgrounds and behaviors in relationship to performance in 6.002 x. In *In the Sixth Learning Int. Networks Consortium conf.*., 2013.

[10] R. Ferguson and D. Clow. Examining engagement: analysing learner subpopulations in massive open online courses (MOOCs). In *the Fifth Int. conf. on Learning Analytics And Knowledge ACM*, pages 51–58, 2015.

[11] M. Fire, G. Katz, Y. Elovici, B. Shapira, and L. Rokach. Predicting student exam's scores by analyzing social network data. In *the 7th Int. Workshop on Active Media Technology*, 2012.

[12] P. J. Guo and K. Reinecke. Demographic differences in how students navigate through MOOCs. pages 21–30, 2014.

[13] G. Hofstede and G. J. Hofstede. *Cultures and organizations: Software of the mind (3rd ed.)*. McGraw-Hill, New York, USA, 2010.

[14] G. Hofstede, G. J. Hofstede, M. Minkov, and H. Vinken. *Values survey module 2008.* URL: http://www. geerthofstede. nl/media/253/VSM08English. doc, 2008.

[15] A. N. Hunt and S. Tickner. Cultural dimensions of learning in online teacher education courses. *Journal of Open, Flexible and Distance Learning*, 19(2):25–47, 2015.

[16] K. J. Kim and C. J. Bonk. Cross-cultural comparisons of online collaboration. *Journal of Computer-Mediated Communication*, 8(1), 2002.

[17] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: Analyzing learner subpopulations in massive open online courses. In *the 3rd Int. conf. on learning analytics and knowledge*, pages 170–179, 2013.

[18] F. K. Leung. In search of an East Asian identity in mathematics education. *Educational Studies in Mathematics*, 17(1), 35-51, 2001.

[19] S. O. Nesterko, S. Dotsenko, Q. Han, D. Seaton, J. Reich, I. Chuang, and A. D. Ho. Evaluating the geographic data in MOOCs. In *the 2013 conf. on Neural Information Processing Systems*, 2013.

[20] C. G. Northcutt, A. D. Ho, and I. L. Chuang. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Assoc.*, 47(260):583–621, 1952.

[21] A. Ogan, E. Walker, R. Baker, M. M. T. Rodrigo, J. C. Soriano, and M. J. Castro. Towards understanding how to assess help-seeking behavior across cultures. *Int. Journal of Artificial Intelligence in Education*, 25(2):229–248, 2015.

[22] A. Ogan, E. Yarzebinski, P. Fernández, and I. Casas. Cognitive tutor use in Chile: Understanding classroom and lab culture. In *the 17th Int. conf. on Artificial Intelligence in Education*, pages 318–327, 2015.

[23] A. C. Ordóñez. *Predicting Int. Critical Success Factors in e-learning: A comparison of four universities from China, Mexico, Spain and USA*. PhD thesis, Universitat Oberta de Catalunya, August 2014.

[24] P. Parrish and J. Linder-VanBerschot. Cultural dimensions of learning: Addressing the challenges of multicultural instruction. *The Int. Review of Research in Open and Distributed Learning*, 11(2), 2010.

[25] M. M. T. Rodrigo, R. S. J. D. Baker, and L. Rossi. Student off-task behavior in computer-based learning in the Philippines: comparison to prior research in the usa. *Teachers College Record*, 115(10):1–27, 2013.

[26] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[27] M. Saarela and T. Kärkkäinen. Do country stereotypes exist in PISA? a clustering approach for large, sparse, and weighted data. In *The 8th Int. conf. on Educational Data Mining*, 2015.

[28] Y. Wang and R. Baker. Content or platform: Why do students complete MOOCs? *MERLOT Journal of Online Learning and Teaching*, 11(1), 17-30, 2015.

[29] M. Wen and C. P. Rosé. Identifying latent study habits by mining learner behavior patterns in massive open online courses. In *the 23rd ACM Int. conf. on conf. on Information and Knowledge Management*, pages 1983–1986, 2014.

[30] J. K. K. Wong. Are the learning styles of Asian int. students culturally or contextually based? *Int. Education Journal*, 4(4), 154-166, 2004.

[31] M. Zhu, Y. Bergner, Y. Zhang, R. Baker, E. Wang, and L. Paquette. Longitudinal engagement, performance, and social connectivity: a MOOC case study using exponential random graph models. In *The 6th Int. conf. on Learning Analytics and Knowledge*, 2016.

Effect of student ability and question difficulty on duration

Yijun Ma
Teachers College,
Columbia University
New York, NY 10027
ym2476@tc.columbia.edu

Lalitha Agnihotri
McGraw Hill Education
2 Penn Plaza
New York, NY 10121
lalitha.agnihotri@mheducation.com

Ryan Baker
Teachers College,
Columbia University
New York, NY 10027
ryanshaunbaker@gmail.com

Shirin Mojarad
McGraw Hill Education
281 Summer Street
Boston, MA 02210
shirin.mojarad@mheducation.com

ABSTRACT

Time has become a standard feature used in EDM models, and is used in models of meta-cognitive strategies to models of disengagement. Most of these models consider whether a student action is "too fast" or "too slow". However, an open question remains on how we define and select these cut-offs. Moreover, it is not clear that the same cut-offs are appropriate across different situations. Some students may generally respond faster than others; more difficult items may take different amounts of time. In this paper, we consider whether absolute or relative indicators of time are more appropriate as cut-offs, and whether simple transformations (such as log time) are useful when representing time. We do so through visualizing student performance in relation to general student ability, item difficulty, and different ways of representing time. We find that student knowledge and item difficulty should be taken into account when choosing cut-offs, and that there are advantages to representing duration in terms of standardized log-time.

Keywords

Time taken, Duration, Visualization, Student Ability, Rasch Model, 1PL, Item Difficulty

1. INTRODUCTION

Over the decade since the Educational Data Mining community began to coalesce, one of the most common ways to interpret student behavior has been to look at the amount of time taken to respond to questions. Early work by Aleven, Baker, and Beck tried to determine whether a response was "too fast", indicating gaming the system, help abuse, try-step abuse, or disengaged behavior [1, 2, 3]. Soon, work began to consider whether a response was "too slow" as well [4]. Researchers noted that performance seemed to degrade when behavior reached either of these two extremes. This theme of trying to identify behavior as "too fast" or "too slow" continues to this day [5, 6]. Actions that are "too fast" or "too slow" are seen as components in a range of EDM models, including contemporary models of gaming the system [7], off-task behavior [8, 9], carelessness [10, 11], and self-explanation [12].

However, one of the interesting aspects of this body of literature is how remarkably inconsistent it is, as noted by [5]. Despite their conceptual simplicity, researchers do not agree what "too fast" or "too slow" means. This inconsistency may not be a major concern when these parameters are empirically fit using training labels, but is somewhat more concerning when cut-offs are rationally defined.

Part of the reason for inconsistency, of course, is that "too fast" and "too slow" are inherently contextual. Interfaces matter. A student completing division problems by typing in answers is likely to respond faster than a student chasing down a skeleton and hitting the right divisor key [13]. Ability matters. A 7-year old solving arithmetic problems is likely to perform more slowly than a 38-year old. Difficulty matters. Even for the same user interface and an experienced adult, "49 / 7" will be solved more quickly than "602 / 7".

For this reason, it is unlikely there is a universal answer to how fast is too fast, and how slow is too slow. Nor will it be easy to find a simple formula or set of formulas that can predict this. Mathematical models based on memory [14] can make predictions about speed in some situations, but are incomplete for many of the complex types of problem-solving and the activities surrounding problem-solving in modern learning environments. At the same time, there exist simple psychometric models that can predict a considerable amount of variance in performance, which may be useful in investigations of this nature.

One solution, as discussed above, is to empirically select a single cut-off, but part of the challenge is that even within a learning environment, cut-offs both vary contextually, and exist on a continuum. In this paper, we will examine this continuum in a visual fashion, across different situations within a single online learning environment. Specifically, we will analyze how the relationship between time and performance varies when students vary in knowledge, and for items of different overall difficulty.

We will also investigate whether the most commonly used way to represent time (number of seconds) is the best representation for understanding these issues, or whether standardizing or transforming time makes it easier to understand the relationship between time and performance.

By better understanding these relationships, we will be able to select more appropriate cut-offs, and develop more precise models for discovery with analysis and interventions.

2. DATA SET

We investigate these issues in the context of one of the world's most widely used digital learning environments, McGraw-Hill Education's Connect system [16, 17]. Connect is currently actively used by approximately two million students and 25,000 instructors. Within Connect, instructors select questions from question banks and the system then administers them to the student as homework, quiz, exam, or practice assignments. Most items are auto-graded by the system, and immediate feedback is provided when relevant (e.g. not during exams). Within homework and practice assignments, students can make multiple attempts to answer each question, based on the policies set up by the instructor. In this paper, we use item and questions interchangeably.

Connect is organized into courses; each course is tied to a McGraw-Hill book title, and question banks are organized in relation to book chapters. In this paper, we focus on a single textbook in order to avoid including radically different material together in the same analysis (for example, one might expect calculus problems to take longer to solve than questions about the factual aspects of history). We analyze a data set from 173 courses that utilize the title *McGraw-Hill's Taxation of Individuals and Business Entities, 6th Edition,* by Brian Spilker, a medium-sized data set with relatively consistent item design, involving a course text with items selected as a focus for enhancement within McGraw-Hill at the time this research was being conducted. Within this textbook, there were multiple types of items: multiple choice items where single responses were correct, multiple choice items where multiple responses were correct, fill-in-the-blank items, matching questions, and ungraded essays (removed prior to analysis).

Within this textbook, within the period between August 2014 and November 2014, 3,882 students (working with 86 instructors) answered 2,947 distinct questions. In total, this set of students attempted to answer questions 536,520 times, an average of 138.21 attempts per student.

Prior to analysis, we removed all ungraded questions from the data set (as assessing correctness is outside the scope of this paper). We also removed attempts where the student timed-out due to inactivity within the system for 60 minutes, and where the student's response time was not collected or had impossible values (due to logging errors). For this specific analysis, we removed students' second and subsequent attempts to answer questions, focusing on their performance and time taken on their first attempt. Although second and subsequent attempts are relevant to issues of modeling student behaviors such as off-task behavior and gaming the system, these times are strongly influenced by the time taken on the first attempt, and are relatively more complex to consider. As such, we leave analysis of second and subsequent attempts to future work. The resultant cleaned data set involved 3,632 students answering 2,689 distinct questions, attempting to answer items 365,302 times, an average of 100.58 attempts per student.

Within these items, scores were distributed between 0 and 1, with 76% of items receiving a fully correct score of 1. However, partial credit was assigned by instructors and, as a result, is somewhat non-uniform; different items had different partial credit assigned for different responses. As such, the partial credit information was less useful for analysis than in other systems where it is assigned in a consistent fashion [15, 18]. To avoid having our results impacted by this inconsistency, we assigned a value of 0 (incorrect) to any student response that was not fully correct. Only 7.9% of the problem attempts were affected by this modification.

2.1 Tagging with Question Difficulty and Student Ability

In order to understand how student knowledge and item difficulty influence the relationship between time taken and performance, we annotated the data with a well-known psychometric model: the Rasch Model [19, 20].

The Rasch Model is one of the most widely used models in the history of psychometrics. It relates performance to student ability (treated here as overall knowledge of the domain) and item difficulty. More recent and advanced models from the psychometrics and student modeling literature consider change in knowledge over time, group items into latent skills, explicitly model the probability of guess and slip, and use different uncertainty functions for students and items [21, 22, 23, 24, 25]. However, the Rasch model is appropriate for the analysis here, as assesses student knowledge and item difficulty (which is what we focus on in the analyses below), it is known to function well when different students answer different items [19], and has high stability and reliability [20].

The equation for the Rasch model is given as follows [19]:

$$P(\theta) = \frac{1}{1+e^{-1(\theta-b)}}$$

where b is the question difficulty parameter, θ is the student ability (knowledge) level, and $P(\theta)$ is the probability that the student will answer the current item correctly. Within this model, if a student's ability is equal to the item's difficulty ($\theta = b$), the probability that the student will answer the question correctly is 50%. As the student's ability becomes higher or the item's difficulty becomes lower, the probability of correctness increases and finally is approximately equal to 1; correspondingly, as ability becomes lower or difficulty becomes higher, the probability of correctness approaches 0.

As is standard [19], we use Maximum Likelihood Estimation, in this case converging after seven iterations, to estimate the values of θ and b for each student and item based on actual data. After fitting and applying the model, all student attempts are tagged with a difficulty parameter and an ability parameter.

This model achieves an R-squared value of 0.322, and an A' (mathematically equivalent to AUC but easier to calculate) of 0.852, calculated using the A' calculator available at http://www.columbia.edu/~rsb2162/computeAPrime.zip .

3. Analysis

We analyze the research questions discussed above through a set of visualizations, created in Python's matplotlib library. Each of the visualizations will place some variant of the time taken by the student to give a response on the X axis, and place the percentage of times when the student response was correct (percent correct) on the Y axis. In the visualizations, item responses are binned to one-second grain-size. For that

bin, we find the percent correct and plot a dot there; if there are more items in the bin, the dot is made larger.

3.1 Baseline Graph

In the first visualization, Figure 1, we consider the baseline relationship between time taken and percent correct. Item difficulty according to the Rasch model is also included in the visualization as color, with darker colored dots representing easier items and lighter dots representing harder items (e.g. if a dot is dark, the items composing that dot were on average easier).[12]

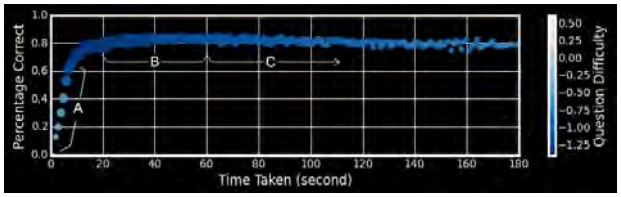


Figure 1: The relationship between the time taken to respond to an item, and correctness. Color is used to denote item difficulty.

As Figure 1 shows, students who spend very little time on an item typically achieve low percentage correct. As the time taken increases, performance improves, curving up from 0 seconds to about 12 seconds; this range of the graph is denoted "A". Percent correct remains stable from 20 seconds to 60 seconds; this range of the graph is denoted "B". As students spend over 60 seconds, their performance somewhat declines again; this range of the graph is denoted "C". This graph shows a similar qualitative pattern to the pattern seen in other systems, but with the shifts occurring at different points. For example, Beck [3] finds that performance improves up until the student has spent 4 seconds, remains stable under 7 seconds, and drops gradually after that.

It is worth noting that despite these shifts, it is non-trivial to find cut-offs. 12 seconds is approximately the inflection point where performance shifts to being stable, but it probably contains more positive behavior than would be desired. It might still be desirable to pick a lower cut-off point for "too fast". Similarly, the difference between 60 seconds and 100 seconds for "too long" is relatively minimal.

One limitation to Figure 1 is that fewer and fewer data points are seen as the times get longer, making it difficult to show all the data in a relatively limited horizontal space. This limitation can be addressed by switching from absolute time in seconds, to a logarithmic scale for time, shown in Figure 2. By switching to a logarithmic scale, the long tail of long response times is compressed to a small section of the plot and we can show more data while maintaining the essence of the graph. The log scale thus makes it easier to present our full data.

The log scale also makes it easier to see that there are more inflection points than Figure 1 showed. The same ranges (0-12 seconds, 20-60 seconds and 60+ seconds) are marked in Figure 2 as in Figure 1, to enable comparison. Note that between 0-12 seconds (range A), there is a secondary inflection point around 3.5 log time taken where performance shifts from improving slowly to improving quickly. This might be a better cut-off for "too fast" than 12 seconds. Similarly, the decline in performance can be seen to begin around 4.75 log time taken but to accelerate after 5.5 log time taken, suggesting a potentially better "too slow" cut-off. While these cut-offs are somewhat harder for a reader to interpret directly from the numbers, they allow us to make more sophisticated distinctions than were possible just from absolute time.

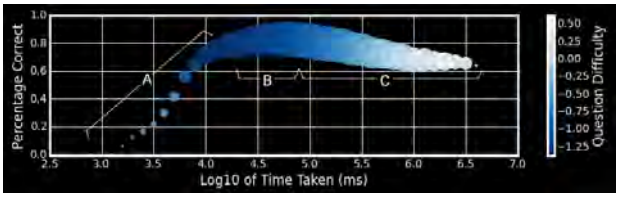


Figure 2: The relationship between the time taken (log scale) to respond to an item, and correctness. Color is used to denote item difficulty.

3.2 Standardization

One common decision seen in many models that measure student time [26, 27] is to represent student time in terms of standard deviations faster or slower than the average time, calculated as a Z-score, and referred to as standardized time or unitized time. This transformation, which assumes that time is normally distributed, uses the formula

$$Z = \frac{Time - Mean(Time)}{SD\ (Time)}$$

The logic is that this approach accounts for the fact that different items need different amounts of time to answer them, allowing fairer comparison of student time on different items.

Figure 3 shows the results of applying this transformation to our data.

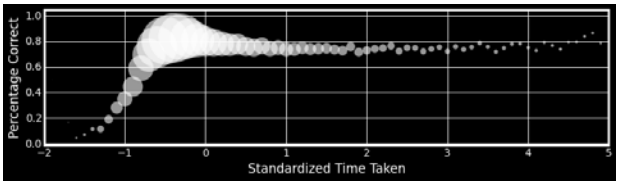


Figure 3: The relationship between the standardized time taken to respond to an item, and correctness.

As this graph shows, most of the data is now clumped together. Notably, the center of the data is not at 0 SD; instead the median is somewhere around -0.5 SD. Though 0 SD is by definition the average value, it is clearly not the median value. This is a common limitation to using standardization, and one that the authors have observed in previous data sets as well. As such, using standardization is vulnerable to skewness and outliers in the original data, making it broadly unsuitable for use across data sets – or indeed, for cases where the magnitude of the long time outliers may vary over time. This can occur, for example, when the original data set has a small number of students with extremely high outlier times, or when the system time-out may change over time. This suggests that standardized time is undesirable for use in cut-offs, since the cut-off points may vary depending on the exact outliers in the data set. This could be addressed by ignoring the outliers when computing

the SD value (i.e. truncating the values of extreme outliers [28];) but doing so will only incompletely address a second problem; the data is highly compressed relative to the previous visualizations we have examined. Most of the data points occur in a fairly small range. In this case, 64.4% of the data is clumped between Z= -1 and Z = 0. If the data were distributed according to assumptions, 68% of data would be clumped between Z= -1 and Z= 1, double the range. This clumping makes it difficult to see the inflections in performance for rapid student responses; although the graph's clumping does allow us to see that there is some rise in performance for very high response times (a set of outliers outside of bounds for the earlier representations).

One alternative, shown in Figure 4, is to use [29] modified Z-score, which is computed as:

$$M_i = \frac{0.6745\ (Time - Median(Time))}{MAD\ (Time)}$$

where MAD stands for Median Absolute Deviation.

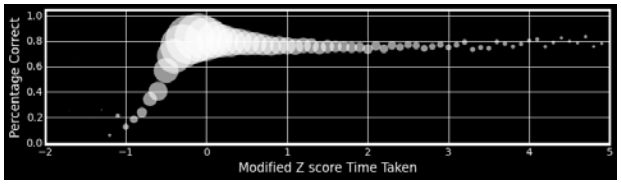


Figure 4: The relationship between the modified Z-score standardized time taken to respond to an item, and correctness.

This approach centers the data better, but does not solve the problem of the data being compressed.

Another alternative is to conduct standardization on time transformed to a logarithmic scale, shown in Figure 5. As we saw in the previous section, using a logarithmic scale spread out the data better and allowed us to see inflection points more clearly.

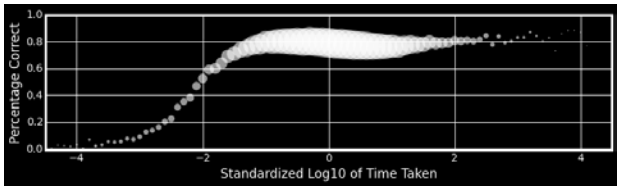


Figure 5: The relationship between the standardized log-transformed time taken to respond to an item, and correctness.

As Figure 5 shows, standardizing using a logarithmic scale centers the data as well as using modified Z-score, but spreads the data out better. The data is broadly centered on Z = 0, with most of the data (68.82%) between Z = -1 and Z = 1 (almost exactly the amount that one would expect for normally distributed data). The same inflection points visible at the left side of Figure 2 are visible at the left side of Figure 5. At the same time, while the logarithmic nature of the transformation does compress the right tail somewhat, we nonetheless can see the same rise in performance at very high time taken that we saw in Figure 3. As such, this representation helps us in understanding the data and choosing cut-offs, while gaining the benefit of comparability that standardizing variables gives us.

3.3 Studying Item Difficulty

One factor that is worth considering is that the time taken appears to be associated with how difficult the items are. Figures 1 and 2 each show difficulty in terms of color, with blue representing easier items (according to the Rasch model discussed above) and white representing harder items.

In Figure 1, we can see that the hardest items are found at the two ends of the spectrum; the briefest times taken, and the longest times taken. It is unsurprising that students take longer on hard items. The connection between difficulty and brief responses is also reasonable; students are more likely to become disengaged and engage in behaviors such as gaming the system and carelessness when encountering hard items [30]. The same pattern is seen in Figure 2, although whether the lowest difficulty is seen for higher or lower times varies between graphs. This is simply a result of the fact that Figure 2 shows more of the data set than Figure 1, due to the use of a logarithmic scale.

This leads to the question of how we should expect the relationship between the student's time taken and their performance to change based on item difficulty. In particular, does the same amount of time taken mean different things for easy items versus difficult items? It is plausible to hypothesize – for example – that rapid responses on easy items may imply fluent knowledge [31] but rapid responses on difficult items may imply disengagement [3].

We examine this by grouping items, based on their difficulty according to the Rasch model *b* parameters, into 5 bands, shown in Table 1, and displayed in Figures 6 and 7.

Table 1: The difficulty groups shown in Figures 6 and 7, based on *b* in the Rasch model. Items with *b* below -3 look very similar to items with *b* from -1 to -3, so they are included in the same group.

Difficulty < -1	Dark Blue
Difficulty 0 to -1	Light Blue
Difficulty 0 to 1	Light Yellow
Difficulty 1 to 3	Yellow
Difficulty > 3	Red

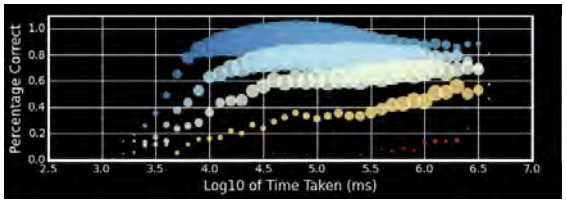


Figure 6: The relationship between the log-transformed time taken to respond to an item, and correctness, for each of the difficulty bands shown in Table 1.

As Figure 6 shows, the pattern for dark blue and light blue (the lower-difficulty items) is largely the same as in Figure 2. Correctness increases fairly rapidly when students spend more time, leveling off and then slowly declining for high amounts of time spent. However, the amount of time needed for high

levels of correctness is higher for the light blue items (*b* between 0 and -1) than for the dark blue items (*b* below -1). This suggests that the same cut-off for "too fast" is not appropriate for items with different difficulty.

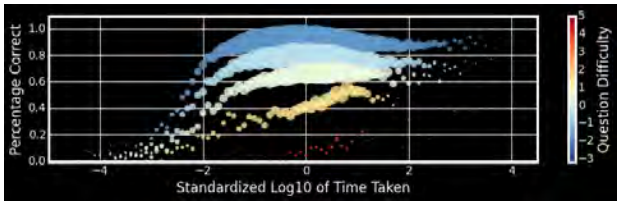


Figure 7: The relationship between the standardized log-transformed time taken to respond to an item, and correctness, for each of the difficulty bands shown in Table 1.

As Figure 7 indicates, this difference between the time needed for the lowest-difficulty items (dark blue) and the moderately low-difficulty items (light blue) cannot be controlled for, simply by switching to standardized log time. Even after we switch to standardized log time, more time is needed for the moderately low-difficulty items than for the lowest-difficulty items, to reach high levels of correctness.

The decline in performance for students who spend too much time (possibly going off-task, or asking for help) is seen for both of these two item difficulty groups, in both the log-time graph and the standardized log-time graph.

Interestingly, the patterns seen are different for the higher-difficulty items. Focusing on yellow and red, we can see that there is no clear inflection point where spending more time is associated with worse performance, or even a clear leveling off in performance. For yellow (*b* between 1 and 3), there is a range between -1 and -1.5 standardized log time where performance may be leveling off or mildly dropping, but it is at best a minor and brief shift, compared to the lower-difficulty bands. For yellow, "too fast" cut-offs could be placed within the -1 to -1.5 SD range, somewhat higher than for lower difficulty (it is hard to identify any good place for a cut-off in the non-standardized graph). For red (*b* above 3), there is essentially no range where increasing time does not improve performance. For neither of these bands is there a clear "too slow" range, where performance worsens once too high a time spent is reached.

These graphs show that time cut-offs should not be considered independently of item difficulty. We are not aware of any models of gaming the system, carelessness, off-task behavior, or related constructs that explicitly consider item difficulty. Our results suggest that this omission is lowering the quality of these models.

3.4 Studying Student Knowledge

Finally, we consider how the student's knowledge of the domain impacts their time spent. Figures 8 and 9 each show knowledge in terms of color, with green representing more knowledgeable students (according to the Rasch model discussed above) and white representing less knowledgeable students. Note that this color scheme corresponds to the color scheme used for difficulty – students are less likely to produce correct answers for white dots.

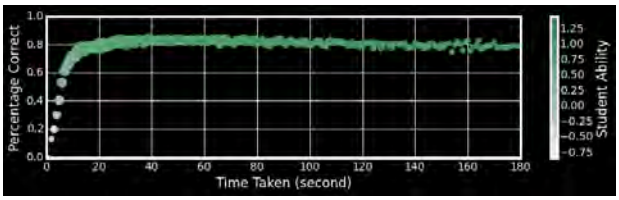


Figure 8: The relationship between the time taken to respond to an item, and correctness. Color is used to denote student overall domain knowledge, assessed using the ability parameter in the Rasch model.

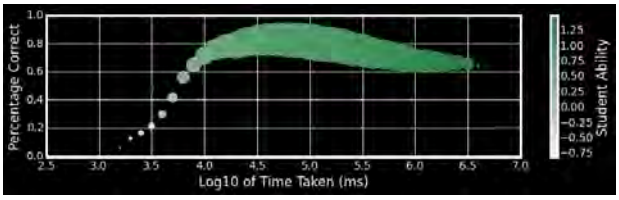


Figure 9: The relationship between the log transformed time taken to respond to an item, and correctness. Color is used to denote student overall domain knowledge, assessed using the ability parameter in the Rasch model.

Figures 8 and 9 show a different pattern than Figures 1 and 2. Whereas those earlier figures indicated that short and long times were seen for hard items, Figures 8 and 9 indicate that brief times are seen for the least able students while long times are generally seen for knowledgeable students. This result suggests that less knowledgeable students appear to be more likely to engage in behaviors such as gaming the system and carelessness, but there does not seem to be a similar pattern for off-task behavior.

Figure 10 shows the same item difficulty bands as were seen in Figure 7, but colored in terms of student ability rather than item difficulty. We can see that regardless of question difficulty, if the response time is too fast relative to the average for the item, the student is likely to be of low ability. However, we can also see from box T1 that this low ability is also seen for longer response times for harder items. For the easiest items, lower ability is seen below -2 SD for time; for the hardest items, lower ability is seen below -1.2 SD for time. As such, this figure indicates that the behavior of answering too fast is seen across questions with different difficulties, though the cut-off should differ.

For higher difficulty items, longer time taken is associated with better students, as shown in T2. But this effect only manifests for the higher difficulty items; these items are more discriminative in terms of the relationship between student ability and longer time taken. Finally, most of the examples of responses that are relatively much longer than other responses occur on the easier items – it is harder to distinguish responses that are genuinely too long for harder items.

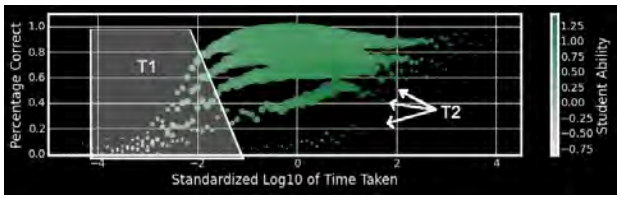


Figure 10: The relationship between the log-transformed time taken to respond to an item, and correctness, for each of the difficulty bands shown in Table 1, but colorized in terms of student ability.

Given these results, we can reasonably ask: how should we expect the relationship between the student's time taken and their performance to change based on the student's general knowledge of the item? In particular, does the same amount of time taken mean different things for knowledgeable students versus not knowledgeable students? Correspondingly, with the above, it is plausible to hypothesize – for example – that rapid responses by knowledgeable students may imply fluent knowledge but rapid responses by struggling students may imply disengagement [14].

We examine this by grouping students, based on their knowledge level according to the Rasch model θ parameters, into 5 bands, shown in Table 2, and displayed in Figure 11.

Table 2: The difficulty groups shown in Figure 11, based on b in the Rasch model. Items with θ below -3 look very similar to items with θ from -1 to -3, so they are included in the same group.

Knowledge < -1	Dark Red
Knowledge 0 to -1	Brick Red
Knowledge 0 to 1	Pink
Knowledge 1 to 3	Light Green
Knowledge > 3	Green

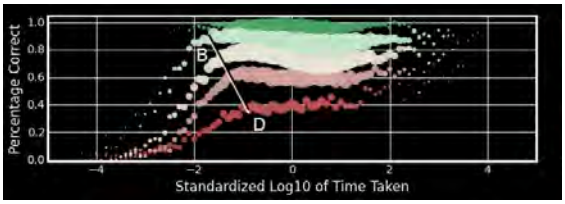


Figure 11: The relationship between the standardized log-transformed time taken to respond to an item, and correctness, for each of the student knowledge bands shown in Table 2.

As Figure 11 shows, the pattern for brick red, pink, and light green (the medium-knowledge students) is largely the same as in Figure 9. Correctness increases fairly rapidly when students spend more time, leveling off, declining, and then coming back up a little for the highest amounts of time spent. The pattern is different for the highest-knowledge students.

The highest-knowledge students (green) essentially do not have any very rapid responses and show similarly high performance across the spectrum of time taken. This can be interpreted in at least three ways. Perhaps the highest-knowledge students do not become disengaged; alternatively, perhaps the students who never become disengaged perform better, and appear to have the highest knowledge. Or perhaps being classified by the Rasch model as having the highest knowledge requires both having the highest knowledge and never becoming disengaged.

The lowest-knowledge students (dark red) have very poor performance for low amounts of time spent. However, their performance never flattens out, although the rate of improvement slows. The more time these students spend, the better they do. Despite that, these students' performance never reaches a very high level.

One other thing that is visible in the graph is that the amount of time needed for asymptotic levels of correctness is lower for the higher knowledge students (θ above 1) than for the lower knowledge students (θ below 0). See the line B-D in the Figure, which links the asymptotic point for high-knowledge students to the near-asymptotic point for low-knowledge students. This suggests that the same cut-off for "too fast" is not appropriate for students with different ability.

4. DISCUSSION AND CONCLUSIONS

In this paper, we have investigated how the relationship between the time taken by students and their performance is mediated by student general knowledge and item difficulty. We also investigate whether different ways of representing time (standardized or non-standardized; log-transformed or non-transformed) impact our ability to recognize cut-offs and inflections in student performance. We analyze these questions by visualizing the relationship between time taken and performance under each of these different conditions.

We find that using a logarithmic scale allows for showing more data while making it easy to present the full data range while standardization allows for a fairer comparison of student time on different items. We find that the combination of these approaches facilitates identifying cut-offs and infection points in student performance.

We find that students who spend very little time on an item typically achieve low percent correct and as the time taken increases, performance improves. However, as students spend over a certain time, their performance somewhat declines again. The amount of time needed for very successful performance is different for easier and harder items and is higher for the easy items compared to very easy items. Hence, we suggest that the same cut-off for "too fast" is not appropriate for items with different difficulty levels.

Student performance declines when students spend too much time on easy and very easy items. The patterns seen are different for the higher-difficulty items. For the difficult and very difficult items, we do not observe any clear inflection point where spending more time is associated with worse performance.

As such, we can conclude that time cut-offs should not be considered independently of item difficulty. We are not aware of any models of gaming the system, carelessness, off-task behavior, or related constructs that explicitly consider item difficulty. Our results suggest that this omission is lowering the quality of these models.

In terms of student overall domain knowledge, we find that the most successful students seldom respond in very short amounts of time. As discussed above, this may reflect in part the fact

that very quick responses make the student appear generally less successful within the Rasch model. However, we also see that the generally knowledgeable students show consistently high performance for most the span of time taken, whereas the less generally knowledgeable students' performance does not level off to the same degree.

For higher difficulty items, longer time taken is associated with better students. However, this effect only manifests for the higher difficulty items; these items are more discriminative in terms of the relationship between student ability and longer times taken. In future work, we will try to correlate these longer times with students' usage of other online materials during. At present we do not have access to this level of detailed data.

These results suggest overall that models that consider student time taken during online learning, and select time cut-offs, should take student general knowledge and item difficulty into account. However, the exact cut-offs will probably differ between systems and also possibly differ with content.

It would be useful to investigate whether the findings seen here are general across other contexts. In our future work, we will investigate their generality to other textbooks, and whether the findings also generalize to other online learning platforms. It would also be useful to examine existing models depending on time cutoffs, and see whether measures of general student knowledge (perhaps average correctness so far across skills) and item difficulty can produce more accurate models of constructs like gaming the system and off-task behavior. Ultimately, this type of model may enhance the effectiveness of behavior detection, leading to more effective interventions to struggling and disengaged students. One of our upcoming steps will be to use these analyses to develop behavior detectors for our platform, that can be used to help to students who are answering too fast or who are struggling and responding slowly. We will then measure the impact of these changes on learning outcomes, to see the degree to which these approaches can enhance student learning.

5. ACKNOWLEDGMENTS

Our most sincere thanks to Mark Riedesel and Alfred Essa for supporting this research. We would also like to thank Malcolm Duncan and his team at the EZTest who helped us navigate through the database and helped create queries to pull the data required for this research.

6. REFERENCES

[1] Aleven, V., McLaren, B., Roll, I. and Koedinger, K., 2004, August. Toward tutoring help seeking. In Intelligent Tutoring Systems (pp. 227-239). Springer Berlin Heidelberg.

[2] Baker, R.S., Corbett, A.T. and Koedinger, K.R., 2004, August. Detecting student misuse of intelligent tutoring systems. In Intelligent tutoring systems (pp. 531-540). Springer Berlin Heidelberg.

[3] Beck, J. E., 2005. Engagement tracing: using response times to model student disengagement. In Artificial intelligence in education: Supporting learning through intelligent and socially informed technology, 125, p.88-95.

[4] Baker, R.S., 2007, April. Modeling and understanding students' off-task behavior in intelligent tutoring systems. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 1059-1068). ACM.

[5] Kovanović, V., Gašević, D., Dawson, S., Joksimović, S., Baker, R.S. and Hatala, M., 2015, March. Penetrating the black box of time-on-task estimation. In Proceedings of the Fifth International Conference on Learning Analytics And Knowledge (pp. 184-193). ACM.

[6] Muldner, K., Burleson, W., Van de Sande, B. and VanLehn, K., 2010, June. An analysis of gaming behaviors in an intelligent tutoring system. In Intelligent Tutoring Systems (pp. 184-193). Springer Berlin Heidelberg.

[7] Paquette, L., de Carvalho, A.M.J.A. and Ryan, S.B., 2014, July. Towards understanding export coding of student disengagement in online learning. In Proc. of the 36th Annual Cognitive Science Conference (pp. 1126-1131).

[8] Cetintas, S., Si, L., Xin, Y.P. and Hord, C., 2010. Automatic detection of off-task behaviors in intelligent tutoring systems with machine learning techniques. Learning Technologies, IEEE Transactions on, 3(3), pp.228-236.

[9] Pardos, Z.A., Baker, R.S., San Pedro, M.O., Gowda, S.M. and Gowda, S.M., 2013, April. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In Proceedings of the Third International Conference on Learning Analytics and Knowledge (pp. 117-124). ACM.

[10] San Pedro, M.O.C.Z., d Baker, R.S. and Rodrigo, M.M.T., 2011, June. Detecting carelessness through contextual estimation of slip probabilities among students using an intelligent tutor for mathematics. In Artificial Intelligence in Education (pp. 304-311). Springer Berlin Heidelberg.

[11] Hershkovitz, A., de Baker, R.S.J., Gobert, J., Wixon, M. and Sao Pedro, M., 2013. Discovery With Models A Case Study on Carelessness in Computer-Based Science Inquiry. American Behavioral Scientist, 57(10), pp.1480-1499.

[12] Shih, B., Koedinger, K.R. and Scheines, R., 2011. A response time model for bottom-out hints as worked examples. Handbook of educational data mining, pp.201-212.

[13] Habgood, M.J. and Ainsworth, S.E., 2011. Motivating children to learn effectively: Exploring the value of intrinsic integration in educational games. The Journal of the Learning Sciences, 20(2), pp.169-206.

[14] Pavlik, P.I. and Anderson, J.R., 2005. Practice and Forgetting Effects on Vocabulary Memory: An Activation - Based Model of the Spacing Effect. Cognitive Science, 29(4), pp.559-586.

[15] Wang, Y. and Heffernan, N., 2013, July. Extending knowledge tracing to allow partial credit: Using continuous versus binary nodes. In Artificial Intelligence in Education (pp. 181-188). Springer Berlin Heidelberg.

[16] Feild, J., 2015. Improving student performance using nudge analytics.Educational Data Mining.

[17] Agnihotri, L., Aghababyan, A., Mojarad, S., Riedesel, M. and Essa, A., 2015. Mining Login Data For Actionable Student Insight. In Proc. 8th International Conference on Educational Data Mining.

[18] Bridgeman, S., Goodrich, M.T., Kobourov, S.G. and Tamassia, R., 2000. PILOT: An interactive tool for learning and grading. ACM SIGCSE Bulletin, 32(1), pp.139-143.

[19] Baker, F.B., 2001. The basics of item response theory. For full text: http://ericae.net/irt/baker

[20] Bond, T. and Fox, C.M., 2015. Applying the Rasch model: Fundamental measurement in the human sciences. Routledge.

[21] Corbett, A.T. and Anderson, J.R., 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction, 4(4), pp.253-278.

[22] Pavlik Jr, P.I., Cen, H. and Koedinger, K.R., 2009. Performance Factors Analysis--A New Alternative to Knowledge Tracing. Online Submission.

[23] Junker, B.W. and Sijtsma, K., 2001. Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. Applied Psychological Measurement, 25(3), pp.258-272.

[24] Haberman, S.J., 2006. An elementary test of the normal 2PL model against the normal 3PL alternative. ETS Research Report Series, 2006(1), pp.i-8.

[25] Pelánek, R., 2014. Application of Time Decay Functions and the Elo System in Student Modeling. Proc. of Educational Data Mining, pp.21-27.

[26] Baker, R. S., Corbett, A. T., Roll, I., & Koedinger, K. R. (2008). Developing a generalizable detector of when students game the system. User Modeling and User-Adapted Interaction, 18(3), 287-314.

[27] San Pedro, M.O.Z., d Baker, R.S. and Rodrigo, M.M.T., 2014. Carelessness and affect in an intelligent tutoring system for mathematics. International Journal of Artificial Intelligence in Education, 24(2), pp.189-210.

[28] BUlrich, R. and Miller, J., 1994. Effects of truncation on reaction time analysis. Journal of Experimental Psychology: General, 123(1), p.34.

[29] Iglewicz, B. and Hoaglin, D.C., 1993. How to detect and handle outliers (Vol. 16). Asq Press.

[30] Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A. and Koedinger, K., 2008. Why students engage in" gaming the system" behavior in interactive learning environments. Journal of Interactive Learning Research, 19(2), p.185.

[31] Mettler, E., Massey, C.M. and Kellman, P.J., 2011. Improving Adaptive Learning Technology through the Use of Response Times. Grantee Submission.

Modeling the Influence of Format and Depth during Effortful Retrieval Practice

Jaclyn K. Maass
University of Memphis
Institute for Intelligent Systems
Memphis, TN 38152
1-352-584-9103
jkmaass@memphis.edu

Philip I. Pavlik Jr.
University of Memphis
Institute for Intelligent Systems
Memphis, TN 38152
1-901-678-2326
ppavlik@memphis.edu

ABSTRACT
This research combines work in memory, retrieval practice, and depth of processing research. This work aims to identify how the format and depth of a retrieval practice item can be manipulated to increase the effort required to successfully recall or formulate an answer, with the hypothesis that if the effort required to answer an item is increased there will be more benefit to learning. This hypothesis stems from work on desirable difficulties and the effortful retrieval hypothesis. Our data source was an experiment that used a 2 (question depth: factual, applied) x 2 (answer format: multiple choice, short answer) between-subjects design to investigate the effects of these conditions on retrieval practice performance. The experiment was delivered online though Mechanical Turk (n = 178). A logistic regression predicting performance during practice indicates that participants get more (in terms of an increase in future predicted success) from successful retrievals of items that fall within the more difficult level of both the format and depth factors (i.e., short answer and applied). There is also some support that the benefit from multiple choice items may be increased by asking deeper, more applied questions. The application of these results to scheduling effective practice is discussed.

Keywords
Retrieval practice, application, difficulty, multiple choice, short answer, modeling, depth of processing

1. INTRODUCTION
The testing effect is the well-replicated benefit of retrieval practice (i.e., "testing yourself"), typically over the course of several repetitions [e.g., 1; 7; 16; 30; 33]. Experiments often compare the benefit of active retrieval against re-reading or re-studying written material and much of the early work in this field utilized a more traditional cognitive psychology experimental setup (e.g., using word lists/pairs or isolated facts, controlling for prior knowledge, and post testing with verbatim items repeated from practice). This design, however, does not well represent how retrieval practice would be implemented in authentic educational settings. For implementation in classrooms, issues that have real-world importance to educators, such as the format of the questions and the ease of administration, should be considered.

The effect of answer format has long been of interest not only to educational researchers (e.g., comparing multiple choice, fil-in-the-blank, essays, etc.), but also to cognitive psychologists (e.g., comparing recognition, cued or free recall, etc.). Research has shown a continuum in terms of performance/difficulty ranging from recognition, to cued recall, to free recall which translates roughly in educational terms to multiple choice, short answer, and essay questions. This ordering is found consistently in research and is summed up nicely by Glover's [13] work which reported the effectiveness of three formats used during retrieval practice (referred to as intervening tests): free recall, cued recall, and recognition (see Experiment 4). After reading a passage and having intervening tests in one of the three formats, participants took a retention test after four days. The free recall intervening test was an open-ended format, with participants writing what they remembered from the passage. The cued recall intervening test was a fill-in-the-blank format, using sentences paraphrased from the text. The recognition intervening tests required the participants to identify which of several sentences they had read previously in the text. The final retention test included items in each of the three formats (across the posttests in Experiments 4a, 4b, and 4c). A very clear pattern emerged: the fewer cues there were available during practice (e.g. free recall provided the fewest cues), the better participants performed on the final retention test. Those who had intervening tests in a free recall format out-performed participants in the cued recall condition on the final retention test (statistically significant difference), who in turn outperformed those who practiced with a recognition task (not statistically significant). Perhaps most importantly, this advantage held regardless of the format of the retention test, which included all three formats [13].

There are several other studies which show us the benefit of using fewer cues (e.g., short answer format) during retrieval practice. Kang, McDermott, and Roediger III [18] had participants read several journal articles. After reading each article, participants completed one of four possible tasks- a multiple choice test, a short answer test, reading relevant facts from the text, or a questionnaire (i.e., filler task). When feedback was provided during the practice tests, those items that had been practiced in short answer format had significantly higher scores on the final test. Results also indicated that practice with multiple choice testing was no better than re-reading relevant facts. The researchers concluded with a recommendation for practice testing with short answer items. Similar results were found in work by McDaniel, Anderson, Derbish, and Morrisette [22], which indicated that weekly practice tests were more effective in increasing final test performance when the weekly practice was in the form of short answer questions compared to multiple choice items. Since the final test was only in multiple choice format, it suggests another benefit of short answer

practice is the ability to overcome transfer-appropriate-processing effects, which would predict that the final test performance would be highest when it matched the conditions of earlier practice [24]. In other words, short answer may be a better alternative to multiple choice regardless of how you assess it.

One possible reason for why practice with short answer often outperforms multiple choice on final outcome measures is the amount of effort required for retrieval [18]. This general benefit of effortful retrieval has been referred to as the retrieval effort hypothesis, which was motivated by Bjork's [4; 5] desirable difficulty framework and Craik and Lockhart's [11] depth of processing research. The retrieval effort hypothesis, as defined by Pyc and Rawson [29], claims that there is more memorial benefit from successful retrieval practice when it is difficult than when it is less difficult. This follows from the desirable difficulty framework, which suggests that practice which is made more difficult (up to a certain point) will lead to more durable and generalizable learning [4]. The desirable difficulty framework sets a theoretical upper bound on the level of difficulty appropriate for effective learning, which can depend on several individual differences including prior knowledge and working memory capacity. This is similar to the assistance dilemma [19], which suggests there is an optimal middle-ground in terms of how difficult a task should be, and/or how much assistance should be offered to a student during a learning task.

The goal of the current work was to generate data to further investigate the effect of effortful retrieval practice, and specifically, how we can equate the effort required to successfully answer multiple choice items with the effort required for short answer items. One way to address this is to increase the effort required to correctly answer a multiple choice question, and the way to do so may lie within the depth of processing required to respond to the question. By asking a deeper, more applied question, rather than the more common text-based factual question, perhaps we can encourage deeper processing so as to increase the effort required for multiple choice questions.

The depth of processing framework suggests that information which is processed on a deeper level will be encoded in a more elaborate and durable manner, with depth referring to greater semantic or cognitive processing [11]. Craik [10] further defines depth as "the qualitative type of processing carried out on the stimulus…" (p. 307). Questions that require more cognitive processing to successfully answer have also been referred to as deep-reasoning questions. Deep-reasoning questions rely on a student's logic and reasoning abilities and are thought to tap into more complete and coherent understanding [14]. Deep-reasoning questions are embedded in the deeper levels of cognition in Bloom's [6] taxonomy, and both have been shown to be positively correlated with final examination scores [14]. In the current work we attempt to increase the difficulty of multiple choice items by asking deeper, more applied questions, and mine our data to compare the benefit that we see from these more difficult multiple choice items with typical benefit from asking factual short answer items.

The interaction of answer format and depth of processing has been investigated to some degree in work by Smith and Karpicke [31], which compared three answer format conditions :multiple choice, short answer, and hybrid conditions which consisted of short answer-multiple choice pairings. Question type during retrieval practice (i.e., factual and inference questions) was a within-subjects factor (Experiments 1, 2, and 3), but this factor was collapsed in the analyses of final assessment performance. They concluded that practice with short answer could lead to higher performance on the final assessment (compared to practice with multiple choice questions), if students achieve a high proportion of correct short answer responses during practice. Smith and Karpicke therefore attempted to equate the initial practice performance between the short answer and multiple choice conditions. Those results are discussed in more detail in their paper [31], but of importance to the current work is that they attempted to raise performance on short answer questions up to the performance on multiple choice items. The current work will essentially attempt the opposite-increasing the difficulty (or lowering the performance) of multiple choice in an attempt to "equate" it to short answer. Therefore, the design of the current data collection was partially inspired by that of Smith and Karpicke, in an attempt to get more fine-grained information about the interaction between format and depth during practice, and their effect on different format and depths at posttest.

In theory, the multiple choice questions in Smith and Karpicke's work were more difficult when the multiple choice was an inference item, rather than factual, but the nature of their inference questions appears to be fairly straightforward, without requiring much more effort than the factual questions. Specifically, the inference items required participants to combine different facts they had previously read in order to draw a conclusion/answer that had not been explicit in the text. However, for most (if not all) of the inference items, the facts required to answer them were presented within a single paragraph. This is not inherently problematic, but it is important to take note of if your objective is to increase the effort required to answer a multiple choice item, since it brings into question the level of difficulty of the inference questions. For example, an inference would be more difficult to make if it required retrieving and combining more than two facts, or if those facts were presented further apart from each other in the text. Further, the answer options in Smith and Karpicke's multiple choice items only included a single option that appeared in the text- the correct answer option. Thus, these questions become purely a measure of memory (of a previously read text), rather than understanding or learning. In other words, the students wind up asking themselves, "Which of these answer options did I see in/ matches with the text I read earlier" rather than, "Which of these options make sense and accurately reflects what I read?" This only serves to further reduce the difficulty of multiple choice practice. To alleviate this, the multiple choice answer options for the current work were all feasible, text related answers that underwent several iterations, described in detail in the materials section.

1.1 The Current Study

The current study focuses on two ways to increase the difficulty of retrieval: through the amount of retrieval cues available (i.e., the answer format: multiple choice or short answer) and through the depth of processing required to successfully answer the question itself (i.e., the question depth: factual or applied). We attempt to mine our data to determine whether or not the difficulty of multiple choice be increased by asking a deeper question, and whether difficulty created through varying amounts of retrieval cues (i.e., the answer format) is similar to the difficulty created through the depth of the question.

The purpose of this paper is to investigate the effect of question format, depth, and individual differences during retrieval practice. Although the experiment tested several types of transfer at the posttest (e.g., format, depth, and unpracticed information), this paper is predominantly focused on dissecting the mechanisms at play during practice. In order to do this, we employed a method of model-based discovery [3] in which previously developed models are adapted to fit the particular research questions and data being

mined. In order to create a more complete picture, however, some descriptive information regarding posttest performance is provided, although it is not the main focus of this paper.

2. METHODS

2.1 Design

The experiment manipulated difficulty of retrieval practice through a 2 (question depth: factual, applied) x 2 (answer format: multiple choice, short answer) between-subjects design. The difficulty of the posttest was also manipulated with a 2 (posttest question depth: factual, applied) x 2 (posttest answer format: multiple choice, short answer) x 2 (concepts: practiced, unpracticed) fully factorial within-subjects design. This resulted in four between-subjects retrieval practice conditions (Factual MC, Applied MC, Factual SA, or Applied SA), and posttest questions in each of those four conditions, allowing for measures of transfer to a different depth and format, as well as transfer to previously unpracticed concepts. Prior knowledge was assessed by a 6-item pretest on factual questions, half randomly assigned per participant to multiple choice and half to short answer format. This experiment did not include a control condition with no retrieval practice. This was a conscious decision since the testing effect is widely accepted as a reliable phenomenon, and the current design allows for a more tractable, and fine-grained investigation of specific components of retrieval practice.

2.2 Participants

One hundred ninety-three participants were recruited through the Mechanical Turk (MTurk) online data collection platform. The only requirements were for the participants to be at least 18 years of age, a native English speaker, from the United States or Canada, and be a reliable MTurk worker. The last requirement was defined as a worker who had completed at least 50 MTurk tasks with at least a 95% approval rate. Data for 10 participants were removed due to the participants having ten or more time-outs during the experiment and five participants' data were removed due to glitches in the system (n=178, 58% male). Within this sample, 45% were in the age range of 26-34 years, 31% were in the age range of 35-54 years, 30% were between 18-25 years, and 4% were between 55-64 years. Most participants reported that their highest level of completed education was "Some college" (37.2%), followed by "High school/ GED" (17.7%), "Graduate degree" (6.6%), and "Less than high school" (<1%). Each MTurk worker was paid $5.00 for participation

2.3 Materials[1]

2.3.1 Text

The experimental text was 995 words in length and pertained to the circulatory system. It was compiled from texts used in previous research [15; 35], and is estimated to be at a Flesch-Kincaid 6th grade reading level (https://readability-score.com).

2.3.2 Factual and Applied Items

Sixteen concepts were extracted from the text to be used for the creation of factual and applied questions. These concepts represent what we believe to be the crucial components in the text, and are aligned with, and expanded from, the factual questions previously used with these materials [23; 35]. The first author, along with another graduate student familiar with this line of research, created a factual and an applied question based on each of the 16 key concepts. The factual versions for the 16 concepts are taken directly from the text. For example, the text states, "The heart is a pump. Its walls are made of thick muscle. They can squeeze (contract) to send blood rushing out." The factual question for this concept asks, "Which component of the circulatory system acts as a pump?" Answer: the heart.

For each of the 16 concepts, we also created an applied question through brainstorming sessions by asking ourselves the questions, "Why is this fact or component important to the circulatory system?" or "What would happen if this component was not functioning properly?" In most of these cases, the 16 applied questions reference the consequence of the factual relationship (described in the text) not holding true. For example, many applied questions require participants to predict outcomes given a certain component not functioning normally. The key principle for the applied questions is that participants must retrieve the necessary fact or facts from memory (presented previously in the text) and apply them in a new way. Importantly, the text only discusses the normally functioning circulatory system, and presents the material at the factual level, without much elaboration. Therefore, the applied questions are not presented explicitly in the text, but can be answered by processing and recombining the facts contained within the text. For the previous example, the concept of the heart acting as a pump, the applied question is, "Why doesn't oxygen rich blood flow directly from the lungs to the rest of the body?" Answer: Because blood requires a pump, the heart, to push it through the body.

2.3.3 Multiple Choice Answer Options

Each question, both factual and applied, required three (incorrect) answer options for the multiple choice format. The incorrect answer options were created based on common misconceptions about the circulatory system. Information on misconceptions was gathered through past research [e.g., 32] and pilot testing (common incorrect responses to the questions in short answer format). Once three answer options (in addition to the correct answer) were created for each of the factual and applied questions, additional pilot testing confirmed that the frequencies of responses for each of the three incorrect answer choices were not substantially different from each other. This method for creating the answer options was specifically done in an attempt to not lessen the effort required to answer a multiple choice item by using answer options that were unrelated or too easy for a participant to exclude as a possible answer.

2.4 Procedure

The experiment consisted of four portions (pretest, reading, retrieval practice, and posttest) within a single session delivered online through Amazon's Mechanical Turk web service using the MoFaCTS online tutoring system (http://mofacts.optimallearning.org/) [27]. The entire experiment took an average of approximately 60 minutes for participants to complete. After obtaining informed consent, participants completed a pretest consisting of six factual questions. For each participant, half of the questions were randomly assigned to short answer format and the other half to multiple choice. These six questions were created from the text in the same way as those for retrieval practice, but did not overlap with the 16 concepts covered in retrieval practice to reduce the possibility of priming. No corrective feedback was given during the pretest.

[1] Experimental materials are available upon request; please contact the first author.

Next, the participants were asked to read the Circulatory System text which was displayed on a single screen (with a scroll bar). For this portion, participants were instructed to not take notes while they read the text. Participants read at their own pace without a time limit. The average time spent reading was approximately seven minutes.

Following the reading portion, participants began retrieval practice. Each participant was randomly assigned to practice with either factual MC, applied MC, factual SA, or applied SA questions. Retrieval practice consisted of eight questions (each representing a different concept covered in the text), repeated four times each. These eight items were randomly selected for each participant from the list of 16 concepts. The order of the eight questions was randomized for each of the four "blocks" of repetition. Corrective feedback was given immediately after participants entered their responses. Correct responses allowed the participant to immediately move on to the next item; incorrect responses were followed by a review period of 10 seconds, during which the correct response was shown on the screen. This feedback procedure not only provided the correct answer for the participant to review, but also provided an incentive for participants to try their best, since correct answers allowed the participant to "skip" the mandatory 10-second review period. In other words, participants would quickly realize that random guessing or poor effort would only increase the length of the experiment.

The final portion of the experiment was the posttest, which was given after a delay of approximately one minute. During this delay the participants were instructed to complete a "current emotion" survey discussed below. The eight concepts studied during retrieval practice were included in the posttest, but each was randomly assigned to be tested in one of the four format/depth conditions. Each participant also answered an additional eight posttest items (two in each of the format/depth conditions) which reference the eight remaining concepts that were not randomly selected for retrieval practice. This allowed us to see how well each practice condition transferred to similar but previously untested material. Each of the 16 posttest questions were presented once, in random order, without corrective feedback.

At three different points in the experiment, participants responded to a set of six "current emotion" questions. The three time-points were: before the retrieval practice to obtain a baseline, immediately after retrieval practice to look for an effect of practice condition on affect, and immediately after posttest to determine if the change in format and depth at posttest had an adverse effect on affect. Specifically, participants were asked to rate on a scale of 1 (Strongly Disagree) to 5 (Strongly Agree) how much they agree with the statement, "Currently, I am feeling _____." This question was asked six times, with a different affect provided in the blank. The six affects were: anxious, bored, confused, discouraged, frustrated, and unfocused/distracted. Demographic information was also collected at the conclusion of the experiment.

2.5 Scoring

All questions were scored immediately by the system and received a score of 1 or 0 (although this value was not explicitly displayed to the participant). MoFaCTS (the online drill-trial problem authoring and deployment platform we used) scored short answer items by matching words in the participants' responses to key terms necessary to answer the question correctly. Pilot testing revealed common (acceptable) synonyms and alternative words that we incorporated into the system to allow for slight variation in what was considered a correct response. For example, the (complete) correct answer for the (factual) question, "Where is the heart located in relation to the lungs?" is "The heart is located between the lungs." The system scored the responses to this item based on whether or not it contained the word "between" or "middle." The use of regular expressions embedded in the MoFaCTS programming allowed for any of the following responses to be counted as correct: "between the lungs", "the heart is between the lungs", or "the heart's in the middle of the lungs." The regular expressions in the system also accounted for ordering when applicable; for example, ordering is essential for the (factual) question, "Which gas do the cells of the body require to function and which gas do they expel as waste?" Participants received corrective feedback (either "Correct" or "Incorrect. The correct response is _____") after each item in the retrieval practice portion, but not during the pretest or posttest.

3. RESULTS AND DISCUSSION

3.1 Overall Performance

Before we discuss the results of mining our retrieval practice data, it may be helpful to review the broader results of the experiment. Table 1 provides an overview of the average scores for the practice (8 items with 4 trials each), the portion of the posttest containing the eight concepts previously practiced, each randomly assigned to one of the four format/depth conditions, (total of 8 trials), and the portion of the posttest which consisted of eight previously unpracticed concepts, each randomly assigned to one of the four format/depth conditions (total of 8 trials).

The average performances during retrieval practice, provided in Table 1, support the general ordering of performance we expected for each condition. Namely, the Factual MC condition was the least difficult, with the highest performance during practice, the Applied SA was the most difficult condition as indicated by the lowest performance during practice, and the Applied MC and Factual SA fall in between in terms of performance during practice. A between-subjects Analysis of Variance (ANOVA) indicated significant differences between the four conditions, $F(3,174) = 28.49$, $p<.001$. Post hoc pairwise comparisons indicate that the only two conditions that are *not* significantly different from each other are the Applied MC and Factual SA conditions ($p = .19$). All other conditions are significantly different from each other (all p's $< .05$).

Table 1. Means and Standard Deviations for Practice and Posttest Performance by Condition

Retrieval Practice Conditions	Average Practice Performance	Average Posttest Scores†	
		Practiced Concepts	Unpracticed Concepts
Factual MC ($n = 46$)	.85 (.12)	.65 (.17)	.45 (.23)
Applied MC ($n = 42$)	.77 (.17)	.70 (.21)	.45 (.24)
Factual SA ($n = 47$)	.73 (.15)	.69 (.14)	.50 (.19)
Applied SA ($n = 43$)	.55 (.18)	.68 (.20)	.47 (.21)

Note: † collapsed across all format/depth posttest conditions. Standard deviations in parentheses.

Table 1 also displays posttest performance for each condition on the eight concepts they had been tested on during practice, as well as on eight concepts they had read about in the text, but had not actively practiced. Between-subjects ANOVA's showed no

significant differences between conditions for performance on either posttest. Note that the drop in performance from practice to the practiced concepts posttest is due to the within-subjects nature of the posttest conditions. In other words, the eight concepts were only practiced in one condition, but were then randomly assigned to be tested in one of the four depth/format conditions in the posttest, meaning that participants had two items in the posttest of practiced concepts that were in a different format, two that were in a different depth, and two that were in a different format and depth. These different types of transfer in the posttest for the practiced and unpracticed concepts therefore resulted in lowered overall performance. Although not significant, we do see that the Factual MC condition was most affected by these transfer items for the posttest on practiced concepts.

While the ANOVA's offer us a broad view of overall performance, in order to truly answer our research questions we will need a finer-grained analysis. Mining our data and creating a model of learning will give us a more in depth look at what is taking place during retrieval practice.

3.2 Modeling Retrieval Practice

A logistic mixed-effects regression was created to model learning during retrieval practice. Since retrieval practice conditions differed in the question depth and answer format factors according to the result above, this model is meant to dissect the differential learning caused by each type of question. The model is based on a Performance Factors Analysis (PFA) where performance is predicted on subsequent trials as a function of the performance on prior trials [26]. Unlike Additive Factors Modeling (AFM) [8], PFA captures prior performance by two parameters, differentiating the effect of prior incorrect (unsuccessful) and correct (successful) trials. We chose to use PFA to separate these components because it would allow us to look into the difference in predictive ability between successful versus unsuccessful prior retrievals. This comparison would indicate if the benefit of retrieval practice is dependent on successful retrieval, or if the mere attempt at retrieval (i.e., incorrect trials) also results in better performance.

Modeling the data included several iterations guided by our hypotheses concerning the effects of format, depth, and prior knowledge. We began with the basic components of a PFA model: two parameters to capture the count of prior correct and incorrect trials. We also included pretest score and a random effect of subject, all of which were significant.

We then added in features we suspected would affect performance based on the cognitive and educational research discussed above, namely, the format and depth of the practiced items. We used one parameter to capture the format of the current item and one to capture the depth of the current item. We also tried adding measures of response time (e.g., time spent reading the text prior to practice, average time spent on all previous trials, and average time spent on previous trials with the specific item, etc.) but none were significant in the model. Next, we added interactions between all factors that had proven significant at that point (e.g., count of prior correct by depth, count of prior incorrect by pretest, depth by format, etc.) Only two of these interactions were significant: count of prior correct by format and count of prior correct by depth, which were retained in the final model. Finally, several measures of affect were added to the model (i.e., the affective score).

The final additions to the model included measures of affect. Remember that our measure of affect consisted of six questions which each used a 5-point Likert-item (1- Strongly Disagree to 5-Strongly Agree) for participants to rate how much they agreed with the statement: "Currently I am feeling _____" for each of the six different affects (anxious, bored, confused, discouraged, frustrated, and unfocused/distracted). Ratings for each of these six affects were collected before and after retrieval practice (and after posttest, but that was not relevant to modeling the learning during practice). We tested the model using six parameters of the affects before practices, and then six parameters to capture the affect after practice. We decided to try to approximate participants affective states during practice by averaging the self-reported levels of affect reported before and after practice. It should be noted that there was not much change in affect from before to after retrieval practice, and each of the three measures (the "before" ratings, the "after" ratings, and the average of the two) performed similarly in the model. Confusion (averaged to capture affect during practice) was the only affect factor that improved the fit of the model. The last step was adding in interactions between this confusion measure and the count or prior correct and incorrect trials, of which only the latter was significant. The final model, summarized in Table 2, retained each of the parameters that achieved significance throughout our modeling process.

The final model had an R^2 of .359, with 5,696 total observations from 178 participants. The *AIC* was 4838.2, the *BIC* was 4904.6, and the Log Likelihood was 4818.2. Table 2 summarizes the fixed effects parameter values of the final model. Not included in Table 2 is the random effect of Participant (*SD* = 0.669). For the format and depth parameters, a value of 0 was assigned to the less difficult level (i.e., MC and Factual) and a value of 1 was assigned to the more difficult level of each factor (i.e., SA and Applied). For each of the parameters involving the count of prior correct or incorrect trials, the log of (1 + the prior count) was taken to account for diminishing marginal returns expected from the power law of practice [25]. Figure 1 also illustrates the fit of the model (left) to the participants' data (right).

Ten runs of a 10-fold cross-validation revealed that the model retained validity when comparing the training folds (R^2 = .336) to the testing folds (R^2 = .329). The CV proportion (training folds R^2 divided by testing folds R^2) for the model indicated that 97.9% of the validity of the model was retained in the held out data.

Table 2. Summary of Fixed Effects for Logistic Regression Model Predicting Future Success

Parameter	Parameter Estimate	*SE*	*Z*-value
Intercept	-0.11	.19	-.56
Pretest	1.95	.30	6.50
Count of Prior Correct	1.82	.16	11.72
Count of Prior Incorrect	1.47	.15	9.88
Format	-1.22	.14	-8.93
Depth	-0.82	.13	-6.06
Prior Correct x Format	1.13	.19	5.93
Prior Correct x Depth	0.36†	.19	1.93
Prior Incorrect x Confusion	-0.18	.05	-3.78

Note: † $p < .05$; all other parameters are significant at the $p < .001$ level. For the Format and Depth parameters, MC and factual are coded as 0, and SA and applied are coded as 1, respectively.

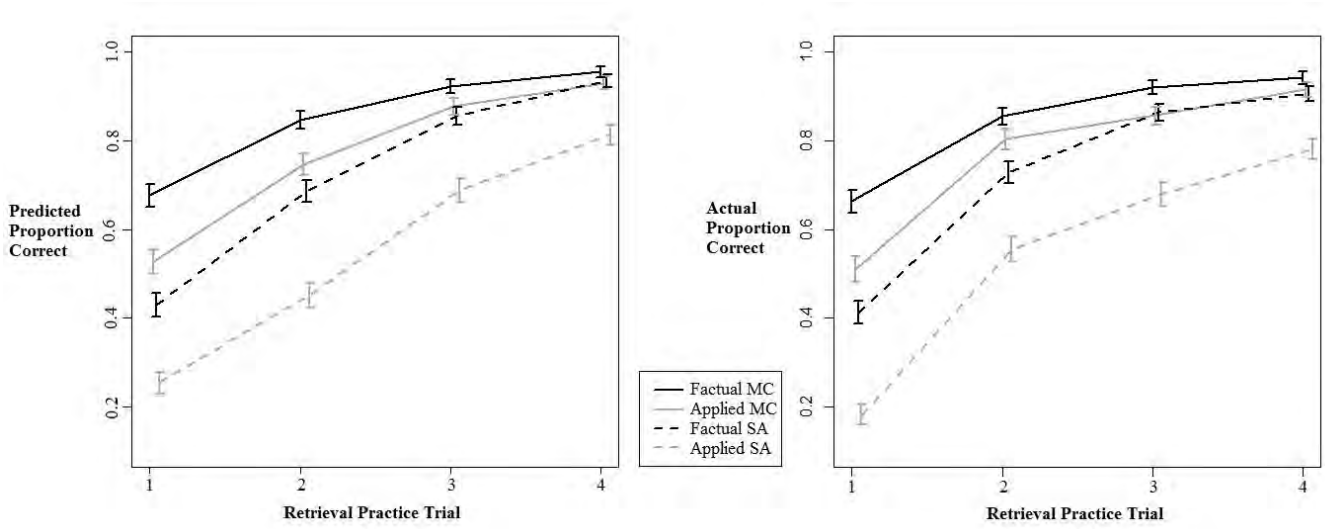


Figure 1. Side by side comparison of the model's predicted performance (left) and the participants' actual performance (right) during the four trials of retrieval practice.

3.3 Model Interpretation

One of the first things the data mining reveals is that correct retrieval (specifically recall) is important for learning. However, the current model also indicates a benefit from unsuccessful retrieval, although to a smaller degree. It is worth noting the model also shows a (lesser) benefit from unsuccessful trials. When comparing just the effect of prior correct and incorrect practice trials, it appears that they offer almost equivalent additions to the prediction/model (1.47 vs 1.82). However, the count of prior correct also interacts with the depth and with the format. For three out of the four practice conditions, these increase the predictive ability of previous successful practices. Therefore, taken all-together, there is much more of a positive effect of previous correct trials than incorrect trials. For example, in the Applied SA condition with one previous correct trial and one previous incorrect trial, successful practices is more than twice as impactful on future performance as previous unsuccessful practices when taking the interactions into account. This difference between the influence from previous correct versus incorrect trials is made even greater if the student has a higher level of confusion (as indicated by the negative estimate for the confusion*incorrect count parameter). This result adds to the building body of research that suggests it is successful retrieval, and not just the attempt to retrieve, that is beneficial to learning [20; 21; 29]. Thus, when it comes to supplying challenging questions for retrieval practice, we must be sure that the questions are at an appropriate difficulty-level for the student, so the student can be successful enough to gain from such practice.

Our model also shows how the format and depth of a practice item influence performance. First we see that the average performance for multiple choice practice is significantly higher than practice with short answer (as indicated by the overall performance of the multiple choice conditions during practice in Table 1 and the -1.22 estimate for short answer practice in Table 2 and), which indicates that multiple choice is the better option in terms of allowing for a higher percentage of successful practice. However, we also saw in the model above that there is more gained from successful short answer practice than is gained from successful multiple choice practice (the Prior Correct x Format parameter). This result are aligned with prior work which suggests that the short answer format may not be universally "better," especially if students are not getting a sufficient amount of those questions correct [31]. Based on these results, it is reasonable to suggest that in order to schedule effective practice, students should be given questions that have a higher likelihood of being answering correctly. If we assume that for the most part, students have a lower level of prior knowledge at the beginning of practice/learning a topic, multiple choice item may permit learning by boosting success. However, since successful short answer practice offers more of a benefit (than multiple choice), it seems that students should eventually transition into short answer practice as they become more proficient. In other words, practice should begin with the less effortful item-type and transition to the more effortful (and more beneficial) item once students reach some level of mastery.

The same may be said for practice with the deeper applied items, over the more text-based factual questions, in that students will get the factual items correct more often, but there is more gained from successful applied practice than from successful factual practice. Again, students might benefit most from beginning with the easier depth (factual/ text-based) and finishing retrieval practice with more difficult, applied questions. The goal it seems, should be to get students to a point where they can get many successful retrieval attempts with SA and/or applied items. This suggestion aligns with ideas in several areas of education research including scaffolding [17], zone of proximal development, and concreteness fading [34]. Determining the optimal level of mastery is an important component though, since increased redundancy during learning (repeated practice of known information) has been shown to offer decreasing marginal returns [9; 28]. Our model also illustrates the importance of taking prior knowledge into account when designing tutoring systems and practice schedules. Some students might be able to begin right away with more difficult items (e.g., applied short answer) and others would benefit from beginning practice with factual multiple choice questions and progress from there.

3.3.1 Affect in the Model

The work concerning affect in the current study is exploratory in nature and was meant to give us an indication of which affective states might be the most important to investigate further in future experiments. Our measure of affective states indicated that the most influential affect was confusion. The interaction between the count

of prior unsuccessful trials and self-reported confusion level in our model shows that when a learner answers more questions incorrectly, higher confusion predicts a much larger negative effect than if a learner has higher confusion but is still having mostly successful practice. This preliminary result appears to align with previous findings which suggest that confusion can be an important component during learning, and is beneficial when students identify that confusion and work to clarify it (i.e. start to produce correct responses), but detrimental when the confusion is overwhelming or the student fails to remedy it [12].

Unlike previous work by Baker, et. al., [2] we did not find any significant impact of frustration or boredom (nor for the other affective states we asked participants about: anxiousness, discouragement, and distractedness). As the current work was meant to serve only as an exploration of affect during retrieval practice, this is an area that we may investigate further in the future. In future work we may implement pop-up/immediate questions concerning the participant's current affective, or specifically their level of confusion, after more than one incorrect response to measure affect/ changes in confusion during bouts of unsuccessful practice.

3.4 General Conclusions

Our model of performance during retrieval practice indicates a benefit for successful retrieval of short answer over multiple choice items. Likewise, there is a benefit from successful retrieval of applied items over factual items which supports the effortful retrieval hypothesis, that successful trials with more difficult items are better than success on less difficult items. Our hypothesis that the difficulty of multiple choice items could be increased (and equated with difficulty of factual short answer items) by asking applied questions, could potentially be supported by the non-significant difference in practice performances, although more analyses will be necessary before making this conclusion. However, format appears to be a more powerful predictor of future success than depth. This may suggest that the difficulty of retrieving information from memory created from less cues (short answer items), is more beneficial than difficulty created through the effortful processing and reasoning with retrieved information (applied items). We recognize that the construct of retrieval effort could be considered too broad of an explanation for our results. While retrieval effort may not capture all the nuances involved in understanding retrieval, we believe it offers a parsimonious general framework under which several mechanisms are captured. Understanding the role that effort plays in retrieval practice will benefit from future work that investigates the differences in more fine-grained mechanisms such as individual difference in strategy use and/or cognitive processes involved in practice with each question type.

4. ACKNOWLEDGMENTS

Our thanks to the University of Memphis's Institute for Intelligent Systems for providing the resources and support necessary to conduct this research.

5. REFERENCES

[1] Agarwal, P.K., Karpicke, J.D., Kang, S.H., Roediger III, H.L., and McDermott, K.B., 2008. Examining the testing effect with open-and closed-book tests. *Applied Cognitive Psychology 22*, 7, 861-876.

[2] Baker, R.S.J.d., D'Mello, S.K., Rodrigo, M.M.T., and Graesser, A.C., 2010. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies 68*, 4, 223-241. DOI= http://dx.doi.org/10.1016/j.ijhcs.2009.12.003.

[3] Baker, R.S.J.d. and Yacef, K., 2009. The State of Educational Data Mining in 2009: A Review and Future Visions. *Journal of Educational Data Mining 1*, 1, 3-17.

[4] Bjork, R.A., 1994. Memory and metamemory considerations in the training of human beings. In *Metacognition: Knowing about knowing*, J.M.A.P. Shimamura Ed. The MIT Press, Cambridge, MA, US, 185-205.

[5] Bjork, R.A., 1999. Assessing our own competence: Heuristics and illusions. In *Attention and performance XVII: Cognitive regulation of performance: Interaction of theory and application*, D.G.A. Koriat Ed. The MIT Press, Cambridge, MA, US, 435-459.

[6] Bloom, B.S., 1956. *Taxonomy of Educational Objectives: The Classification of Education Goals. Cognitive Domain. Handbook 1*. Longman.

[7] Carrier, M. and Pashler, H., 1992. The influence of retrieval on retention. *Memory & Cognition 20*, 6 (Nov), 633-642.

[8] Cen, H., Koedinger, K., and Junker, B., 2008. Comparing Two IRT Models for Conjunctive Skills. In *Intelligent Tutoring Systems: 9th International Conference, ITS 2008, Montreal, Canada, June 23-27, 2008 Proceedings*, B.P. Woolf, E. Aïmeur, R. Nkambou and S. Lajoie Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 796-798. DOI= http://dx.doi.org/10.1007/978-3-540-69132-7_111.

[9] Cen, H., Koedinger, K.R., and Junker, B., 2007. Is Over Practice Necessary? – Improving Learning Efficiency with the Cognitive Tutor through Educational Data Mining. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (Los Angeles, CA2007).

[10] Craik, F.I., 2002. Levels of processing: Past, present . . . and future? *Memory 10*, 5-6 (Sep), 305-318.

[11] Craik, F.I. and Lockhart, R.S., 1972. Levels of processing: A framework for memory research. *Journal of Verbal Learning & Verbal Behavior 11*, 6 (Dec), 671-684.

[12] D'Mello, S., Lehman, B., Pekrun, R., and Graesser, A., 2014. Confusion can be beneficial for learning. *Learning and Instruction 29*, 153-170.

[13] Glover, J.A., 1989. The "testing" phenomenon: Not gone but nearly forgotten. *Journal of Educational Psychology 81*, 3 (Sep), 392-399.

[14] Graesser, A.C. and Person, N.K., 1994. Question asking during tutoring. *American Educational Research Journal 31*, 1, 104-137.

[15] Hathorn, L.G. and Rawson, K.A., 2012. The roles of embedded monitoring requests and questions in improving mental models of computer-based scientific text. *Computers & Education 59*, 3, 1021-1031. DOI= http://dx.doi.org/10.1016/j.compedu.2012.04.014.

[16] Johnson, C.I. and Mayer, R.E., 2009. A testing effect with multimedia learning. *Journal of Educational Psychology 101*, 3, 621.

[17] Kang, H., Thompson, J., and Windschitl, M., 2014. Creating Opportunities for Students to Show What They Know: The Role of Scaffolding in Assessment Tasks. *Science Education 98*, 4, 674-704. DOI= http://dx.doi.org/10.1002/sce.21123.

[18] Kang, S.H., McDermott, K.B., and Roediger III, H.L., 2007. Test format and corrective feedback modify the effect of testing on long-term retention. *European Journal of Cognitive Psychology 19*, 4-5, 528-558.

[19] Koedinger, K.R., Pavlik Jr., P.I., McLaren, B.M., and Aleven, V., 2008. Is it Better to Give than to Receive? The

Assistance Dilemma as a Fundamental Unsolved Problem in the Cognitive Science of Learning and Instruction. In *Proceedings of the 30th Conference of the Cognitive Science Society*, V. Sloutsky, B. Love and K. McRae Eds., Washington, D.C., 2155-2160.

[20] Maass, J.K. and Pavlik Jr, P.I., 2013. Using learner modeling to determine effective conditions of learning for optimal transfer. In *Artificial Intelligence in Education* Springer, 189-198.

[21] Maass, J.K., Pavlik Jr, P.I., and Hua, H., 2015. How Spacing and Variable Retrieval Practice Affect the Learning of Statistics Concepts. In *Proceedings of the 17th International Artificial Intelligence in Education Conference* Springer-Verlag, Berlin, Heidelberg.

[22] McDaniel, M.A., Anderson, J.L., Derbish, M.H., and Morrisette, N., 2007. Testing the testing effect in the classroom. *European Journal of Cognitive Psychology 19*, 4-5, 494-513.

[23] Michael, A.L., Klee, T., Bransford, J.D., and Warren, S.F., 1993. The transition form theory to therapy: Test of two instructional methods. *Applied Cognitive Psychology 7*, 2, 139-153. DOI= http://dx.doi.org/10.1002/acp.2350070206.

[24] Morris, C.D., Bransford, J.D., and Franks, J.J., 1977. Levels of processing versus transfer appropriate processing. *Journal of Verbal Learning and Verbal Behavior 16*, 5, 519-533.

[25] Newell, A. and Rosenbloom, P.S., 1981. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition 1*, 1-55.

[26] Pavlik Jr., P.I., Cen, H., and Koedinger, K.R., 2009. Performance Factors Analysis -- A New Alternative to Knowledge Tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, V. Dimitrova, R. Mizoguchi, B.d. Boulay and A. Graesser Eds., Brighton, England, 531-538.

[27] Pavlik Jr., P.I., Kelly, C., and Maass, J.K., 2016 submitted. Using the Mobile Fact and Concept Training System (MoFaCTS).

[28] Pavlik Jr., P.I., Maass, J.K., and Hua, H., 2015, November. Redundancy causes spacing effects. In *56th Annual Meeting of the Psychonomic Society*, Chicago.

[29] Pyc, M.A. and Rawson, K.A., 2009. Testing the retrieval effort hypothesis: Does greater difficulty correctly recalling information lead to higher levels of memory? *Journal of Memory and Language 60*, 4, 437-447.

[30] Roediger III, H.L. and Karpicke, J.D., 2006. The Power of Testing Memory: Basic Research and Implications for Educational Practice. *Perspectives on Psychological Science 1*, 3, 181-210. DOI= http://dx.doi.org/10.2307/40212166.

[31] Smith, M.A. and Karpicke, J.D., 2014. Retrieval practice with short-answer, multiple-choice, and hybrid tests. *Memory 22*, 7, 784-802.

[32] Sungur, S., Tekkaya, C., and Geban, Ö., 2001. The contribution of conceptual change texts accompanied by concept mapping to students' understanding of the human circulatory system. *School Science and Mathematics 101*, 2, 91-101.

[33] Thompson, C.P., Wenger, S.K., and Bartling, C.A., 1978. How recall facilitates subsequent recall: A reappraisal. *Journal of Experimental Psychology: Human Learning & Memory 4*, 3 (May), 210-221.

[34] Tullis, J.G., Goldstone, R.L., and Hanson, A.J., 2015. Scheduling Scaffolding: The Extent and Arrangement of Assistance During Training Impacts Test Performance. *Journal of Motor Behavior 47*, 5 (2015/09/03), 442-452. DOI= http://dx.doi.org/10.1080/00222895.2015.1008686.

[35] Wolfe, M.B., Schreiner, M., Rehder, B., Laham, D., Foltz, P.W., Kintsch, W., and Landauer, T.K., 1998. Learning from text: Matching readers and texts by latent semantic analysis. *Discourse Processes 25*, 2-3, 309-336.

The Apprentice Learner architecture: Closing the loop between learning theory and educational data

Christopher J. MacLellan
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
cmaclell@cs.cmu.edu

Erik Harpstead
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
eharpste@cs.cmu.edu

Rony Patel
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
rbpatel@andrew.cmu.edu

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
koedinger@cmu.edu

ABSTRACT

While Educational Data Mining research has traditionally emphasized the practical aspects of learner modeling, such as predictive modeling, estimating students knowledge, and informing adaptive instruction, in the current study, we argue that Educational Data Mining can also be used to test and improve our fundamental theories of human learning. Using the Apprentice Learner architecture, a computational theory of learning capable of simulating human behavior in interactive learning environments, we generate two models that embody alternative theories of human learning: (1) that humans perfectly recall previous training during learning and (2) that humans only recall a limited window of experience. We evaluate which of these models is better supported by data from two fractions tutoring systems. In general, we find that the model with a complete memory better fits the data than a model recalling only the previous training experience (data-drive theory development). Additionally, we demonstrate that both models are able to predict student performances, as well as, reproduce the main effects of an experimental paradigm without being trained on student data (theory-driven prediction). These results demonstrate how the Apprentice Learner architecture can be used to close the loop between learning theory and educational data.

1. INTRODUCTION

One branch of Educational Data Mining (EDM) research leverages data to improve our theoretical understanding of how people learn [28, 3]. Analogous to how data from the Large Hadron Collider can be used to gain insights into physical laws, educational data can be used to provide insights into the unobservable mechanisms underlying student learning. Surprisingly, little EDM research has explored this direction, rather, the main trends in research center on how statistical models can be used to perform latent knowledge estimation and domain-structure discovery (i.e., knowledge component discovery) [3]. While these research directions are important, we argue that the availability of educational data makes the EDM community well poised to contribute substantially towards our theoretical understanding of human learning.

Although many of the widely used predictive models of learning, e.g. Bayesian Knowledge Tracing [5], and Additive Factors Model [4], rely on existing theories of human learning, such as the power law of practice [22], researchers rarely apply these models to educational data with the aim of improving the underlying theory of learning. Further, there are a number of barriers to using educational data for this purpose. First, many EDM models are only loose approximations of the theories they are based on. For example, the Additive Factors Model predicts that improvements in human performance will follow a single logistic function, whereas the power law of practice states that the improvements should follow a power function [6]. Second, EDM models do not reflect the current state of learning theory. For example, recent studies of skill acquisition actually suggest that improvements should follows three distinct power functions, one for each phase of cognitive skill acquisition [30], rather than a single logistic function. This disconnect between theories and models makes it difficult to draw inferences about the underlying theories given the fit of models to data. By more tightly connecting our EDM models to theory, we can leverage educational data to improve our understanding of the mechanisms behind human learning and, in turn, use these theories to improve our abilities to predict student behavior.

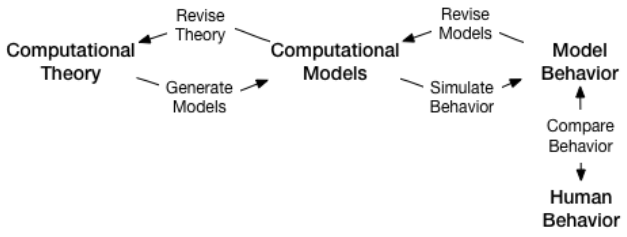


Figure 1: A depiction of how theories, models, and behavior relate. Theories are used to generate models, which can be used to simulate behaviors. Simulated behavior can be compared to human behavior and differences inform future models and theories.

To more tightly link a theory to models, researchers can develop a computational theory [17, 21]. Unlike a theory that only specifies the abstract relationships between constructs (e.g., that an increase in spatial skills leads to an increase in learning with graphical representations [26]), a computational theory represents a complete description of the mechanisms that produce observed phenomena. Within this paradigm, a model presents as a specific algorithmic implementation of these mechanisms that can be executed to simulate behavior, which then can be compared with observed behavior in order to test both the model and the underlying theory. A key component of this approach is not to explain or "fit" a relationship in observed data, but rather, to predict that a relationship will be present before any data is observed. Figure 1 shows the iterative relationship between theories, models, and behaviors. We argue that this approach complements existing approaches in the EDM literature.

In the current work, we present the Apprentice Learner architecture, a computational theory of learning in interactive learning environments, such as tutoring systems. Unlike prior models of student performance, such as Additive Factors Model and its variants, Apprentice Learner models seek to explain the mechanism students use to acquire new knowledge from instruction. This mechanical description allows us to simulate learner behavior within an instructional environment and use these simulations to predict human behavior. Rather than arriving at a general conclusions like students learn differently from positive and negative feedback this approach lets us explore possible explanations for the mechanisms driving these results. In presenting this computational theory we make two claims:

1. The Apprentice Learner architecture can be used to predict student behavior and experimental results before collecting any student data (purely theory-driven prediction).
2. The Apprentice Learner architecture can leverage data to improve learning theory through the creation and testing of different models of learning.

To support these claims, we explore different assumptions about memory and its effect on human learning in intelligent tutoring systems. We leverage the the Apprentice Learner architecture to instantiate two models of human learning, one that hypothesizes perfect memory and another that assumes a more limited window of memory. We apply these models in two different fractions tutoring systems. In both cases, we generate datasets of simulated learner behavior that have high agreement with the patterns of behavior observed in human students. Additionally, we show that our models reproduce the main effects of a problem sequencing experiment without first being fit to student data. In general, we find that the model with perfect memory better fits the fractions data than the model with limited memory; these findings provide an initial demonstration of how our computational theory can be refined in response to data.

In the following sections, we first present the Apprentice Learner architecture and describe the theoretical commitments that it makes. Next, we describe our overarching simulation approach, the particular computational models that we investigate, and the results of our simulation studies in (1) fraction addition and (2) fraction arithmetic. Finally, we discuss the implications of our results and directions for future work.

2. THE PROPOSED ARCHITECTURE

In 2006, VanLehn published his seminal paper describing the step-level behavior of tutoring systems [31]. Although not commonly cited within the EDM literature, VanLehn's description of the general two-loop structure of tutoring systems (i.e., an inner loop for step-level feedback and an outer loop for problem selection) has direct relevance to many recent advances in EDM research. For example, researchers have used knowledge component discovery to create a better understanding of domain tasks [4, 13], so that the inner loop feedback can be improved. Other researchers have used latent knowledge estimation to improve outer loop instructional policies [27]. While VanLehn's theory promotes common ground between similar thrusts of work in EDM, it can only serve as half the picture of a computational theory of the tutoring process.

The Apprentice Learner architecture, shown in Figure 2, is a computational theory of human learning that aligns with the step-level interactions described by VanLehn. The theory embodied in the Apprentice Learner architecture states that students acquire skills by interactively solving problems in a tutored paradigm, receiving correctness feedback on their actions. In the event that the student does not know how to proceed, they can request a hint from the tutor, which provides the student with a demonstration of how to take the next problem-solving step.

The Apprentice Learner architecture uses a base of prior knowledge to induce new skills from its observed demonstrations and feedback. The first kind of knowledge consists of functions for manipulating data (e.g., adding two values, appending two strings together, etc.). The second kind of knowledge consists of features for recognizing different elements in the interface (e.g., recognizing numbers, mathematical symbols, etc.). Depending on the domain, different kinds of background knowledge may be appropriate. For example, Apprentice Learner models in equation solving might have features for recognizing polynomials, whereas models in stoichiometry might have different features for recognizing chemical symbols.

The Apprentice Learner architecture posits three learning mechanisms to induce new skills from prior knowledge and observed demonstrations and feedback. When given a demonstration, the *how* learning mechanism uses function knowledge to search for a sequence of functions that can explain the observed demonstration. After discovering a function sequence, the *where* learning mechanism acquires general perceptual patterns for recognizing the elements used in the discovered sequence. Finally, the *when* learning mechanism uses the tutor state, augmented with feature knowledge, to identify the conditions under which the discovered sequence should be executed. The combination of the components discovered by how, where, and when learning mechanisms constitutes a skill. Apprentice learners apply learned skills in subsequent problem solving.

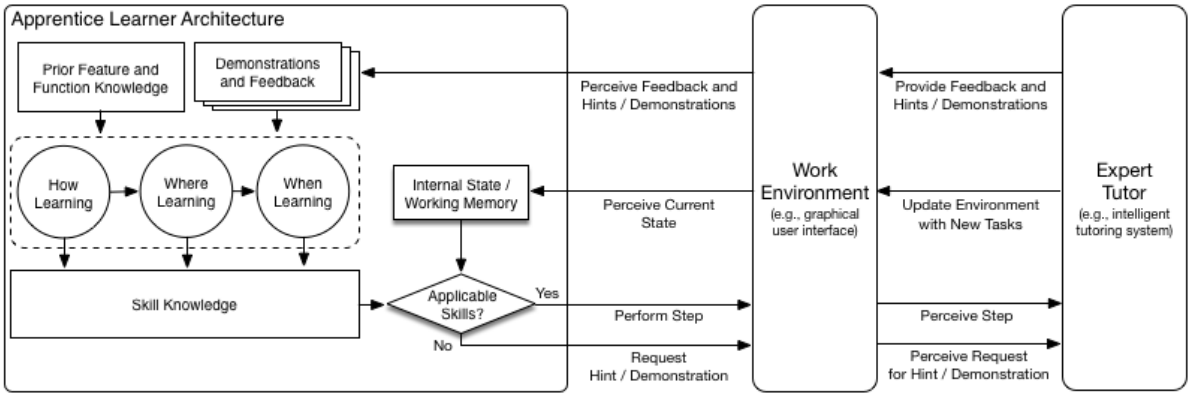


Figure 2: The Apprentice Learner architecture and its interactions between the work environment and expert tutor. The architecture possesses three learning mechanisms (how, where, and when) to generalize demonstrations and feedback into skill knowledge that can be used for problem solving.

In order to apply learned skills, the Apprentice Learner architecture posits that learners use a basic Recognize-Act cycle [32]. When presented with a problem, learners first query their skill knowledge to determine if any known skills are applicable. If an applicable skill is found, the learner executes it. The learner passes correctness feedback on the resulting action to its *when* learning mechanism, which uses the feedback to refine the conditions under which the skill can be executed. In the event that no skills are applicable, the learner requests a demonstration that is passed to the how, where, and when learners to produce a new skill.

Given the computational theory described by our architecture and our data-driven theory development approach (see Figure 1), our goal is to develop a theory that is consistent with available educational datasets, such as those found in DataShop [7] and other similar repositories. To pursue this goal, we propose a research program wherein different models of human learning are generated within the framework of the Apprentice Learner architecture, i.e., specific algorithms are implemented for each of the components of the architecture. These Apprentice Learner models can then be connected to the same intelligent tutoring systems that generated the data found on Datashop. Next, the behavior of these models can be compared to human behavior. Based on the differences between the models and humans, we can revise our theory (e.g., replacing a perfect memory of previous demonstrations and feedback with a memory that only recalls a window of experience), generate new models, and then simulate the revised models to determine if better agreement between models and human behavior can be demonstrated.

3. SIMULATION STUDIES

We make two key claims about the Apprentice Learner architecture: (1) it can be used to predict student behavior without data and (2) it can be used to improve theory by facilitating the exploration of different models. To demonstrate the potential of the architecture and to support our key claims, we conducted simulation studies with two tutoring systems in the domain of fractions [33, 14, 24].

For these simulations, we created an initial model of human learning by implementing each of the components of the Apprentice Learner architecture in computer code. This model was given two features, isPlusSign and isMultSign, which can be used to determine if a string is a plus or multiply sign (i.e., + or ×). It was also given six functions: Add(X,Y), Subtract(X,Y), Multiply(X,Y), Divide(X,Y), CopyPasteString(X), and GenerateCheckMark(). The Add, Subtract, Multiply, and Divide functions returned the result of applying their respective arithmetic operations to their arguments. The CopyPasteString function returns a copy of the string that is passed to it. Finally, the GenerateCheckMark takes no arguments and returns a check mark that can be used to fill checkboxes in the tutor interface. This prior feature and function knowledge represents the basic interface and arithmetic knowledge that students would be expected to know before using a fractions tutor.

Given this prior knowledge, we implemented three machine learning algorithms for the three learning mechanisms outlined in Figure 2. For how learning, we used a variation of Langley's BACON algorithm [9] to discover an explanation of expert demonstrations using the provided functions. For where learning, we used a variation of Mitchell's Version Space algorithm [20] to discover perceptual patterns for recognizing relevant interface elements. Finally, for when learning, we used Quinlan's FOIL algorithm [25] to learn the conditions under which the learned skills can be executed. More details of our algorithmic implementations can be found in previous work, which refers to this particular combination of learning algorithms as the SimStudent model [18, 11].

In this initial model, the skill knowledge acquired from the three learning mechanisms is stored in the form of production rules (i.e., IF-THEN rules). The perceptual patterns learned from the where learner and the conditions acquired by the when learner constitute the IF part of the rule. The function sequence discovered by the how learner constitutes the THEN part of the rule. An example of a human-readable version of a production rule discovered by one of our models might be: **IF** there are two fractions with denomina-

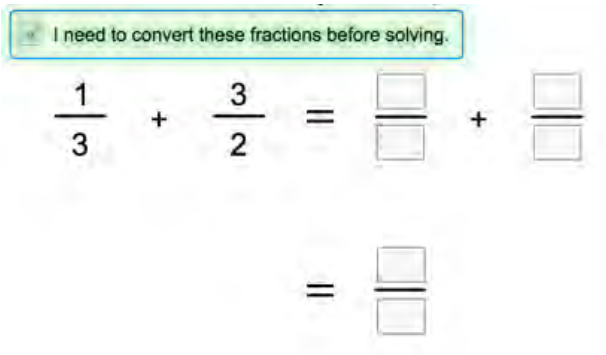


Figure 3: The Fraction Arithmetic Interface.

tors and a sign between them (i.e., the perceptual pattern is present) **AND** the sign is a plus sign and the denominators are equal (i.e., the conditions are satisfied) **THEN** copy one of the denominator values and put the result in the answer denominator box (i.e., perform the function sequence). During problem solving, the models check if they have any applicable production rules (i.e., skills) and if a match is found, then they take the prescribed action.

This initial model, which we refer to as the full-memory model (also the SimStudent model in previous work), has been used to model ordering effects [10], as a teachable agent [18], and for authoring cognitive models [16, 12]. In these previous studies, the full-memory model has been found to regularly outperform human students. One hypothesis is that this model outperforms human students because it revises its skill knowledge using a complete memory of all previous training examples [15]. To explore this hypothesis, we created a second model that duplicates our initial model, with the exception that it only recalls the previous training example during skill learning. This model, which we refer to as the one-back-memory model, instantiates an extreme version of the hypothesis that learners only recall a limited amount of their past experience during learning.

3.1 Data

To test the full-memory and one-back-memory models, we use data from two intelligent tutoring systems available on DataShop. In both tutors, students were asked to solve fraction arithmetic problems using a variation of the interface shown in Figure 3. The first dataset came from the control condition of a fraction addition study [33]. The dataset consisted of 24 students solving 20 fraction addition problems. The tutoring system used in this dataset omitted the "I need to convert these fractions before solving" checkbox and required students to convert fractions to common denominators, even if this meant copying fractions that already had the same denominators. Additionally, the tutor allowed students to use multiple approaches to find a common denominator; they could either multiply the denominators or compute the least common denominator. To allow our models to use this second approach, we added the LeastCommonMultiple(X,Y) function to the prior knowledge of both models, under the assumption that students utilizing this approach know how to compute the least common multiple.

The second dataset came from an experiment testing whether blocking or interleaving different types of fraction arithmetic problems was better for learning [24]. This dataset contains 79 students solving 24 fraction addition problems (10 with same denominators and 14 with different denominators) and 24 fraction multiplication problems. The tutor used in this study required students to check the "I need to convert these fractions before solving" box before making the fields necessary for converting visible. Additionally, on fraction addition problems with different denominators, students were only allowed to compute common denominators by multiplying denominators. Thus, the LeastCommonMultiple(X,Y) function was not included in the models for this dataset.

The experimental manipulation of the second datasets divided students into two conditions, blocked and interleaved. The students in the blocked condition received three blocks of problems: fraction addition problems with same denominators, then fraction addition problems with different denominators, and then fraction multiplication problems. The order of the problems within each block was randomized for each student. In contrast, the students in the interleaved condition received a random ordering of all problems. This experiment showed that students in the blocked condition have a lower overall error rate than students in the interleaved condition. Additionally, the error rates of students in the blocked condition increased when transitioning between different types of problems.

3.2 Method

For each dataset, we tested our full-memory and one-back-memory models of learning by creating instances of each model for each student and connecting these instances to the appropriate tutoring systems. The tutoring systems then tutored the instances through the same order of problems that the respective human students received. In each dataset, we compared the first attempt correctness on each step between the two models and their respective humans. For each model, we computed how often the first attempt correctness agreed with the respective human's first attempt correctness, i.e., accuracy, to quantitatively measure the agreement between model and educational data. We report the mean accuracy and its accompanying 95% confidence interval (95% CI) for each model. Next, for each dataset we plotted overall learning curves comparing the first-attempt performance of the humans to each of the two models. For these learning curves, we used a knowledge component model that labeled each step as exercising a skill corresponding to the field that was updated in the interface. These learning curve graphs demonstrate how the Apprentice Learner architecture can be used to generate theory-driven learning curve predictions. Because each model instance has the same prior knowledge, our simulation studies do not take into account the individual differences in students' prior knowledge. To determine if taking into account student-level effects impacts which model better fits the data, we fit a random-effects logistic regression model with a fixed effect for the model prediction and random effect for the student. We report the Akaike information criterion (AIC) and Bayesian information criterion (BIC) scores to determine which of our two models better fits the data in each case. Note, AIC and BIC values on one dataset are not comparable to the AIC and BIC values on another dataset, they can only be used to

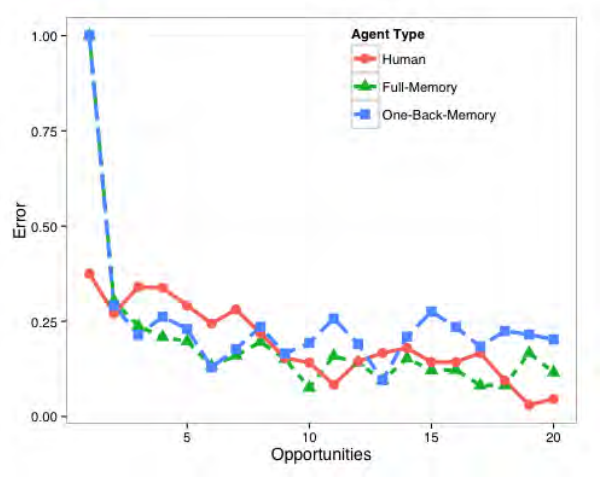


Figure 4: The fraction addition learning curves for the human students, the full-memory model, and the one-back-memory model.

rank model fits on the same dataset. For a given dataset, lower values of AIC and BIC are better, and a difference of more than 3 in either measure is usually viewed as strong evidence to prefer one model over another.

3.3 Fraction Addition Results

After simulating the the 24 students in the fraction addition dataset, we found both models were significantly predictive of students' correctness on the 2,432 first attempts ($p < 0.01$ via a χ^2 test). The full-memory model correctly predicted 74.05% (95% CI : 72.26, 75.79) of first attempts, whereas the one-back-memory model correctly predicted only 70.93% (95% CI : 69.08, 72.73) of first attempts. This significant difference in accuracy ($p < 0.01$ via Mcnemar's test) suggests that the full-memory model more closely agrees with the fraction addition data than the one-back-memory model, when not taking into account differences in students prior knowledge.

Next, we plotted the learning curves comparing both models' performance to the human performance, see Figure 4. The opportunity counts for these learning curves were determined by how many times each student had practiced filling in the relevant interface field (each field is roughly analogous to the skill used to update that field). Both simulated models initially start off without any skills, so their error rate is 100% on the first step. However, the models quickly converge to human-level performance. Although the full-memory model achieves a lower overall error, the one-back memory model appears to have variation that is more equally distributed around the human performance.

To test which model best fits when taking the differences between students' prior knowledge into account, we fit two mixed-effect logistic regression models that had a single fixed effect for the respective simulation prediction (full-memory or one-back-memory) and a random effect for student. We found that the one-back-memory model better fit the student data (AIC=1727, BIC=1744) than that full-memory model (AIC=1754, BIC=1772), suggesting that students in the fraction addition dataset have differences in their overall performance that might correspond to differences in prior knowledge. Further, these results suggest that the one-back memory model better fits student performance when taking these differences into account.

3.4 Fraction Arithmetic Results

Similar to the previous dataset, we found both models were significantly predictive of the 79 students' 18,589 first attempts ($p < 0.01$ via a χ^2 test). We also found that the full-memory model (Accuracy : 84.04%, 95% CI : 83.5, 84.56) was more predictive of students' first attempts than the one-back-memory model (Accuracy : 80.24%, 95% CI : 79.66, 80.81). Similar to our previous fraction addition results, this significant difference in accuracy ($p < 0.01$ via Mcnemar's test) suggests that the full-memory model more closely agrees with the fraction arithmetic data than the one-back-memory model, when not taking into account differences in students prior knowledge.

Figure 5 shows the learning curves comparing the performance of the two models to the human data. Similar to the fraction addition dataset, the opportunity counts for these learning curves were determined by how many times each student had practice filling in the relevant interface field (again, fields are roughly analogous to the skills used to update them). However, in this dataset we plotted separate learning curves for students in the two experimental conditions, blocked and interleaved.

Similar to the fraction addition learning curves, the full-memory and one-back-memory models initially start off with an error rate of 100% on their first steps and quickly converge to human-level performance. However, in this dataset, we can see that both models seem to emulate key differences in the two conditions. First, the human students in the blocked condition have lower error than those in the interleaved condition ($z = -6.136$, $p < 0.01$ via a logistic regression). Both the full-memory ($z = -9.598$, $p < 0.01$) and the one-back-memory ($z = -4.626$, $p < 0.01$) models correctly predict this main effect of condition. Second, the human students in the interleaved condition slowly converge to asymptotic performance, whereas the human students in the blocked condition achieve lower initial error but then have drastic increases in error when transitioning between problem types (e.g., around opportunity 12). The simulated data from both models appears to mirror these effects. While both models experience a spike in error around opportunity 25 when transitioning to multiply problems, the human students, surprisingly, do not show a similar increase. This difference might be explained by the fact that the human students have prior experience multiplying numbers, and fraction multiplication is arguably easier than fraction addition with different denominators (i.e., students have to use multiplication to compute common denominators). In contrast, both the full-memory and one-back-memory models have no experience with multiplication prior to opportunity 25, so they have a 100% initial error on the first multiplication step. This suggests that future work is needed to explore how to populate models with initial training experiences (e.g., teaching the model to do whole-number multiplication before fraction multiplication).

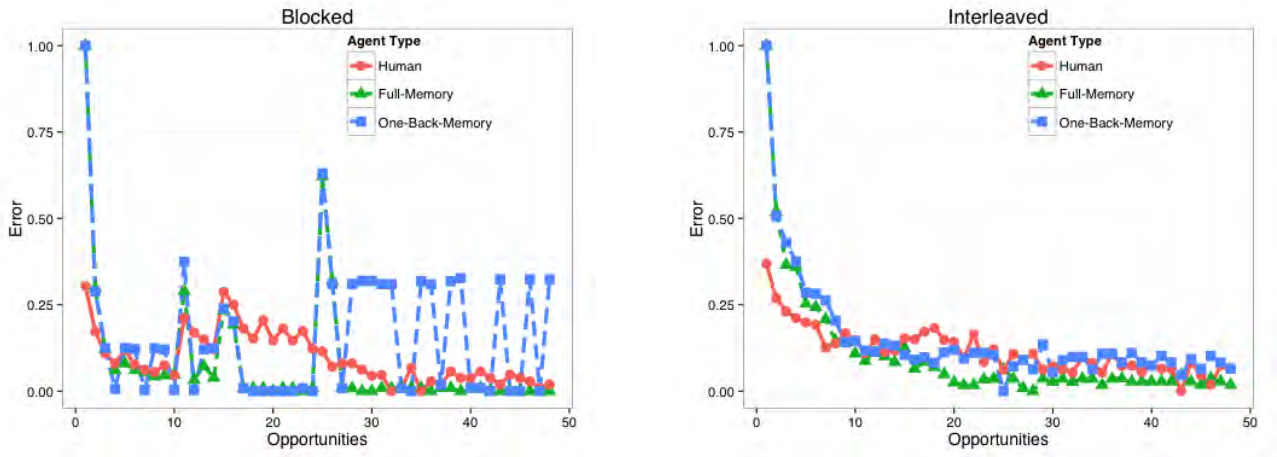


Figure 5: The fraction arithmetic learning curves for the human students, the full-memory model, and the one-back-memory model. The left graph shows the learning curves for the blocked condition and the right graph shows the learning curves for the interleaved condition. The spikes in error rate in the blocked condition occur when students transition from fractions with same denominators to fractions with different denominators (opportunity 12) and to fraction multiplication (opportunity 25).

Finally, we again fit two mixed-effects logistic regression models to determine if taking individual student differences into account would change which of the two models better fit the data. In contrast to the fraction addition results, we found that the full-memory model better fit the student data (AIC=10849, BIC=10872) than that one-back-memory model (AIC=11013, BIC=11036). These results show that, for fraction arithmetic, the full-memory model better fits the student data regardless of whether or not overall student differences are taken into account.

4. GENERAL DISCUSSION

We argue that our simulation studies in fraction addition and fraction arithmetic provide strong evidence in support of our two key claims about the Apprentice Learner architecture. First, our analysis shows that the behavior generated by both models agrees with the human behavior in both fractions datasets; i.e., the full-memory model, which fits best, achieves 75% agreement in the fraction addition dataset and 84% agreement in the fraction arithmetic dataset. Furthermore, we show that both of the models predict the main experimental effect for the fraction arithmetic dataset; i.e., both models correctly predict that the overall performance in the blocked condition will be better than the overall performance in interleaved condition. To our knowledge, these two results are the first example in the EDM literature of how student performance can be precisely predicted in a completely theory-driven way without having to fit the models to the student data first.

Although our models have a reasonably high agreement with the student data, there are still some key differences between the models and the humans. In particular, the models always have 100% first-attempt error on novel skills. While these exaggerated error rates might be useful for detecting transitions between skills (e.g., when using learning curve analysis to develop knowledge-component models [5]), they also suggest an opportunity to improve our underlying theory and models. In future studies we should explore approaches for initializing both prior knowledge (e.g., using students' pretests to choose prior features and functions) and skill knowledge (e.g., pretraining models in a whole-number arithmetic tutor).

Our second key claim was that the Apprentice Learner architecture can be used to improve our underlying theory of human learning using educational data. We argue that our simulation results provide strong evidence supporting this claim. In particular, we tested two different models that operationalize two alternative theories of human learning: the full-memory model, which posits that humans have perfect recall of prior demonstrations and feedback when learning skills, and the one-back-memory model, which is an extreme version of the theory that humans only recall a limited window of prior demonstrations and feedback during skill learning. In our analysis, we showed that the full-memory model better fits both fractions datasets, suggesting that it is a better model of human learning. Next, we used a mixed-effects logistic regression analysis to take into account student differences. Using this approach, we showed that the one-back-memory model better fit on the fraction addition dataset and the full-memory model better fit on the fraction arithmetic dataset.

In general, these results suggest that the full-memory model better fits the fractions datasets than the one-back-memory model (in three out of four cases). However, our results leave open the possibility that, when taking into account overall student differences, a hybrid model might be best (e.g., an n-back model). Further, the full-memory model best fits the educational data, but seems to have better asymptotic performance than the human students. The original inspiration for the one-back-memory model was to decrease this asymptotic performance to bring it into closer alignment with the human performance, but our results suggest that we should consider alternative approaches for decreasing performance.

One possibility would be to replace the when learner with an incremental machine learning algorithm, such as TRESTLE [15]. This approach would let apprentice learners leverage existing theories of interference effects [2] to improve their fit with educational data. In summary, our simulation studies provide strong evidence to support our claims that the Apprentice Learner architecture can be used perform theory-driven prediction and to improve theory based on differences between model and human behavior.

5. FUTURE WORK

The results of our studies have been encouraging, however, we do not wish to leave the impression that the Apprentice Learner architecture is a complete computational theory of learning. Instead, we present the theory as an initial framework that is flexible enough to support new hypotheses about learning. In future work, we plan to explore several variations of the current theoretical structure and invite the community to extend the theory to explain phenomena in their own work.

One affordance of the Apprentice Learner architecture is that it facilitates a search among alternative theories and models. Not unlike existing techniques for searching the space of domain models [4], a search among alternative Apprentice Learner models would let us explore several hypotheses of human learning. For example, it is questionable whether how, where, and when are the correct combination of internal learning mechanisms. It may be that the FOIL algorithm, currently used for when learning, could be used to model both the where and the when learning. This would suggest that the current distinction between where and when learning is artificial and that a single mechanism might produce more human-like simulated data. Alternatively, it could be argued that the architecture is biased by having features provided as prior knowledge, rather than learning features from experience. This argument implies that some mechanism for acquiring new features, effectively a *what* learner, could be included in the architecture [11]. Beyond adding or merging learning mechanisms each individual mechanism could be represented by several underlying algorithms. For example, our implementation of the Version Space algorithm conducts a specific-to-general search for perceptual patterns, but another possible variation would be to conduct a general-to-specific search. Exploring all of these possibilities could be framed as a search task over different parametrizations of the architecture for models that generate the most human-like simulation data.

In the current work, we compare model and human error rates, but the Apprentice Learner architecture allows for finer-grained evaluation. Rather than compare simulated and human learners on whether they performed a step correctly, we could compare learners in terms of their literal response on a step. This opens up the ability to evaluate theories of student misconceptions and how they might affect the particular responses students make [19]. Similarly, in this study we only compared performance on first step attempts, because this is a common convention in EDM, but the high-fidelity simulation data can be used to examine learner behavior beyond the first attempt. Ultimately a unified theory of apprentice learning should account for all of the behaviors learners exhibit on their path to mastery.

As we have stated previously, we view the current state of the Apprentice Learner architecture as incomplete. There are several aspects of learning that the model does not currently account for, such as the effects of delayed feedback [29], the impacts of metacognition [1], and the behavior of collaborative learners [23]. Crucially, however, the theory is not fundamentally incompatible with these ideas. For example, a reinforcement learning paradigm could be employed to back-propagate correctness from delayed feedback. The role of metacognition could be accounted for with a more nuanced variation on the recognize-act cycle that takes into account metacognitive decisions. Finally, instantiating multiple Apprentice Learner models within the same environment and allowing them to generate demonstrations for each other could serve as an initial computational model of collaborative learning. These are just a few examples of how the structure of the architecture can be augmented to incorporate and test additional learning theories.

Finally, in future work we would like to explore how the theoretical tenets of our architecture align with those made by other architectures, such as ACT-R or SOAR [8]. These architectures, which primarily focus on problem solving, have mechanisms for learning skill conditions and for compiling commonly executed sequences of skills into macro-skills. It would be interesting to investigate the extent to which these learning mechanisms align with the when (condition) and how (function sequence) learning mechanisms of the Apprentice Learner architecture. By investigating how these computational theories might be aligned, we hope to provide for learning science, and more generally cognitive science, the kinds of unified theories that have been so successful in physics and the other hard sciences.

6. CONCLUSIONS

In this paper, we have taken the first steps toward a complete computational theory of learning in interactive environments, such as tutoring systems. Not only do we believe that EDM is capable of improving our fundamental theories of learning, but that is uniquely positioned to do so. Using a computational theory approach, it is possible for every tutored learning dataset in the canon of EDM to test and advance learning theories. We hope that other EDM researchers will also see the potential of the Apprentice Learner architecture and the computational theory paradigm, and we look forward to working together to further develop our collective understanding of human learning.

7. ACKNOWLEDGEMENTS

We thank Peggy Tenison for her feedback on earlier versions of this work. We used the "Grounded Feedback Fraction Addition Tutor" and "Fraction Addition and Multiplication" datasets accessed via DataShop (pslcdatashop.org). This work was supported in part by the Department of Education (#R305B090023) and by the National Science Foundation (#SBE-0836012).

8. REFERENCES

[1] V. Aleven, B. M. McLaren, I. Roll, and K. R. Koedinger. Toward Meta-cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*, 16(2):101–130, 2006.

[2] J. R. Anderson and L. M. Reder. The fan effect: New results and new theories. *Journal of Experimental Psychology: General*, 128(2):186–197, June 1999.

[3] R. S. Baker and P. S. Inventado. Educational Data Mining and Learning Analytics. In *Learning Analytics*, pages 61–75. Springer, New York, 2014.

[4] H. Cen, K. R. Koedinger, and B. Junker. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In *Intelligent Tutoring Systems*, pages 164–175, Berlin, 2006.

[5] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1995.

[6] A. Heathcote, S. Brown, and D. J. K. Mewhort. The power law repealed: The case for an exponential law of practice. *Psychonomic Bulletin & Review*, 7(2):185–207, 2000.

[7] K. R. Koedinger, R. S. J. d. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A Data Repository for the EDM community: The PSLC DataShop. In C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. d. Baker, editors, *Handbook of Educational Data Mining*. CRC Press, 2010.

[8] P. Langley, J. E. Laird, and S. Rogers. Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*, 2009.

[9] P. Langley, H. A. Simon, and G. L. Bradshaw. Heuristics for empirical discovery. In L. Bolc, editor, *Computational Models of Learning*. Springer, 1987.

[10] N. Li, W. W. Cohen, and K. R. Koedinger. Problem Order Implications for Learning Transfer. In *Proceedings of the Eleventh International Conference on Intelligent Tutoring Systems*, pages 185–194. Springer Berlin Heidelberg, 2012.

[11] N. Li, N. Matsuda, W. W. Cohen, and K. R. Koedinger. Integrating representation learning and skill learning in a human-like intelligent agent. *Artificial Intelligence*, 219:67–91, 2014.

[12] N. Li, E. Stampfer, W. W. Cohen, and K. R. Koedinger. General and Efficient Cognitive Model Discovery Using a Simulated Student. In *Proceedings of the 35th Annual Meeting of the Cognitive Science Society*, 2013.

[13] R. Liu, K. R. Koedinger, and E. A. McLaughlin. Interpreting Model Discovery and Testing Generalization to a New Dataset. In *Proceedings of the Sixth International Conference on Educational Data Mining*, pages 107–113, 2014.

[14] R. Liu, R. Patel, and K. R. Koedinger. Modeling Common Misconceptions in Learning Process Data. In *Proceedings of the 6th International Conference on Learning Analytics and Knowledge*, 2016.

[15] C. J. MacLellan, E. Harpstead, V. Aleven, and K. R. Koedinger. TRESTLE: Incremental Learning in Structured Domains using Partial Matching and Categorization. In *Proceedings of the Third Conference on Advances in Cognitive Systems*, pages 1–18, 2015.

[16] C. J. MacLellan, K. R. Koedinger, and N. Matsuda. Authoring Tutors with SimStudent: An Evaluation of Efficiency and Model Quality. In *Proceedings of the 12th International Conference on Intelligent Tutoring Systems*, 2014.

[17] D. Marr and T. Poggio. From understanding computation to understanding neural circuitry. 1976.

[18] N. Matsuda, W. W. Cohen, and K. R. Koedinger. Teaching the Teacher: Tutoring SimStudent Leads to More Effective Cognitive Tutor Authoring. *International Journal of Artificial Intelligence in Education*, 25(1):1–34, 2014.

[19] N. Matsuda, A. Lee, W. W. Cohen, and K. R. Koedinger. A Computational Model of How Learner Errors Arise from Weak Prior Knowledge. In *Annual Conference of the Cognitive Science Society*, pages 1288–1293, 2009.

[20] T. M. Mitchell. *Machine Learning*. McGraw-Hill, Boston, 1997.

[21] A. Newell. You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. In *Visual Information Processing*, pages 283–308, 1973.

[22] A. Newell and P. S. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, 1981.

[23] J. K. Olsen, V. Aleven, and N. Rummel. Predicting Student Performance In a Collaborative Learning Environment. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[24] R. Patel, R. Liu, and K. Koedinger. When to Block versus Interleave Practice? Evidence Against Teaching Fraction Addition before Fraction Multiplication. In *Proceedings of the 38th Annual Meeting of the Cognitive Science Society*, Philadelphia, PA, 2016.

[25] J. R. Quinlan and R. M. Cameron-Jones. Induction of logic programs: FOIL and related systems. *New Generation Computing*, 13(3-4):287–312, 1995.

[26] M. A. Rau. Why do the rich get richer? A structural equation model to test how spatial skills affect learning with representations. In *Proceedings of the Eighth International Conference on Educational Data Mining*, 2015.

[27] J. Rollinson and E. Brunskill. From Predictive Models to Instructional Policies. In *Proceedings of the Eighth International Conference on Educational Data Mining*, pages 1–8, 2015.

[28] C. Romero and S. Ventura. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 40(6):601–618, 2010.

[29] R. A. Schmidt and R. A. Bjork. New Conceptualizations of Practice: Common Principles in Three Paradigms Suggest New Concepts for Training. *Psychological Science*, 3(4):207–217, 1992.

[30] C. Tenison and J. R. Anderson. Modeling the Distinct Phases of Skill Acquisition. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 2015.

[31] K. Vanlehn. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.

[32] D. A. Waterman and F. Hayes-Roth. An overview of pattern-directed inference systems. 1978.

[33] E. S. Wiese. *Toward Sense Making with Grounded Feedback*. PhD thesis, Pittsburgh, 2015.

Mining behaviours of students in autograding submission system logs

Jessica McBroom, Bryn Jeffries, Irena Koprinska and Kalina Yacef
School of Information Technologies, University of Sydney, Sydney, NSW 2006, Australia
jmcb6755@uni.sydney.edu.au, {bryn.jeffries, irena.koprinska, kalina.yacef}@sydney.edu.au

ABSTRACT

Effective mining of data from online submission systems offers the potential to improve educational outcomes by identifying student habits and behaviours and their relationship with levels of achievement. In particular, it may assist in identifying students at risk of performing poorly, allowing for early intervention. In this paper we investigate different methods of following the development of student behaviour throughout the semester using online submission system data, and different approaches to analysing this development. We demonstrate the application of these methods to data from a junior computer science course (N=494) and discuss their usefulness in understanding the common behavioural strategies of students in this course and how these develop over time. Finally, we draw links between behaviour in weekly coding tasks and student performance in the final exam and discuss whether these methods could be applicable midway through the semester.

Keywords

Clustering student behaviour; autograding system; assessment and feedback.

1. INTRODUCTION

Autograding submission systems are valuable tools in a modern teaching environment. By automatically assessing a student's submission, feedback can be returned to the student immediately without increasing the burden of marking for the teacher. Students are empowered to repeatedly improve their submission before a final deadline. However, such systems are only likely to improve the student's learning experience if the student allocates time to use feedback for subsequent submissions.

Teachers know from observation that students adopt a range of approaches to learning exercises, especially when outside the classroom. At one extreme, an ideal student will attempt an exercise immediately, and make increasingly better submissions based upon the feedback received. At the other extreme a student may make their first attempt just prior to the submission deadline, leaving no opportunity to improve or even make a decent first attempt. These behaviours, and many in between, may be due to deeply ingrained habits or external factors such as other time commitments. Using online submission systems in our teaching provide us with the opportunity to exploit the historical data of students' attempts. In this work, we investigated techniques of identifying and following the development of student behaviour over the semester, with specific focus on the application of these techniques to a junior computer science course. We were interested in the most common behaviours of students, whether these behaviours changed over time, and relationships between these behaviours and final exam outcomes. We were also interested in how applicable these methods were midway through the semester.

This paper is structured as follows. We first give an overview of the related work on the use of autograding systems and on mining student behaviour in these systems. Section 3 explains the context in which our data was captured. Section 4 is the main part of the paper: it presents our clustering-based approach to detecting and tracking students' behaviours. We finally conclude with a discussion on these different approaches.

2. RELATED WORK

The use of autograding systems in computer science courses have been reported in [1-6], with the majority of studies focusing on analysing the effectiveness of the autograding systems as opposed to understanding student behaviours. Sherman et al. [1] introduced Bottlenose, an autograding system used in a first year programming course in C, and compared the student behaviour on the same assignments when using Bottlenose and when not using it. The results showed that the number of submissions per student per assignment was significantly higher when using the autograding system, which was attributed to students making use of the feedback to improve their programs. Enström et al. [2] developed Kattis, an automated assessment system used at KTH in Sweden for teaching programming and algorithms courses. The use of Kattis resulted in improved student motivation (increased number of submissions) and also in higher student satisfaction in the course evaluation survey. The autograding system Autolab [3] was developed at Carnegie Mellon University and used in a first year programming course in C. Its real-time scoreboard, which shows the class performance on the assessment task, was found to create a healthy competition encouraging students to improve their assignments, and do this quicker.

There has also been some recent work on mining log data from autograding systems [4-6]. Gramoli et al. [4] analysed the impact of autograding and instant feedback using the system PASTA in various computer science courses, from first to fourth year. They found that the instant feedback was beneficial not only for courses focusing on programming but also for courses that use programming as a tool to solve subject specific problems. The relation between the student performance and the chosen programming language and the time when the students start and finish their assignment submissions was also studied. Koprinska et al. [6] investigated whether students at risk of failing in a first year

programming course can be detected early in the semester, using information from three sources: the autograding system PASTA, a discussion board and assessment marks. They built a decision tree that was able to achieve 87% accuracy in predicting the exam mark from information available in the middle of the semester. It was also shown that using the information from the autograding system improved the accuracy, compared to only using the assessment marks. In [5], data from the same sources was used to define the characteristics of high, average- and low-performing students and predict their performance.

More broadly, the related work also includes mining log data from student submissions in computer science courses. Perera et al. [7] analysed behavioural data from online group collaboration logs in a software development project. The goal was to identify patterns and behaviours associated with positive and negative outcomes. Clustering was applied to find similar students and similar teams, and sequential pattern mining was used to extract sequences of frequent events. Student behavioural data from a high school computer science MOOC was analysed by Tomkins et al. [8]. They characterised the performance of high and low achieving students based on the student behaviour in the course and discussion board, and built a predictive model using support vector machines to predict if a student will pass or fail an exam, conducted after the course has finished.

In this paper we extend the previous work on mining log data from autograding systems in computer science courses. Our goal is to study the evolution of student behaviour during the semester, with a view that this could assist in early intervention in future course offerings or provide guidance for course restructuring. We propose different clustering methods and demonstrate their application in the context of a large first year computer science course. We discuss the effectiveness of these techniques for extracting and understanding behavioural patterns, and how these patterns develop over time.

3. DATA

PASTA is an autograding system for computer programming courses developed in our school [9]. Students submit their solution (programming code) to an assessment task. Then PASTA checks this solution by running a set of tests designed by the teacher and provides immediate feedback to the student about the passed and failed tests. Students can then correct their mistakes and resubmit the solution until all tests are passed. PASTA can be configured in different ways - the number of allowed attempts can be limited or unlimited, some tests can be hidden (i.e. not available for immediate feedback, only available after the deadline) and teachers can also add manual comments to complement the automatic feedback. It supports several languages (e.g. Java, C, C++, Python and Matlab) and has been used for various courses – introductory programming, data structures, algorithms, formal languages, artificial intelligence, databases and networks.

PASTA has received positive feedback from students due to the instant feedback and multiple attempts features. Its use has resulted in better student engagement, and also transparent and fair marking as the same tests are used for all students. For each student and task, the PASTA data contains: all submission attempts, the tests that were passed and failed, the time stamps and the mark obtained.

The data used in this paper comes from a junior unit of study on data structures [10], which ran in Semester 2 of 2015 with 494 students enrolled. Students were using PASTA on a weekly basis to submit exercises, over a period of 11 weeks. The exercises were made available just after the lecture related to the topic (say Hashing) and constitute the core material of the tutorials (2 hour computer-based practical sessions, with a ratio of one teacher to 20 students). Each week, one exercise was flagged for assessment and was due the following week, i.e. 12 days after release. The number of attempts allowed was unlimited.

4. ANALYSIS OF STUDENT BEHAVIOUR

There are many ways students work towards their weekly exercises and use PASTA. For instance, students may start early and submit several attempts until their submission is 100% successful; some may start late and have time to submit only once a half-done attempt; others may not submit anything at all; and so on. Our approach to follow students' behaviour on their weekly work is to first cluster behaviours on all submissions, for all students (section 4.1). Then we explore several ways of tracking students' behaviour during the semester (sections 4.2 to 4.5).

4.1 Submission clustering: typical behaviour on one submission

In order to determine the types of approaches students take when completing weekly tasks, we performed a clustering on all the data available. For each given student and week, we created a vector containing information about the student's behaviour on that week's submission. We chose features which related to student submission times as an indication of their approach to the task. We also included features relating to student marks, number of attempts and number of compile errors, which provided an indication of performance. In total there were 5434 vectors (11 weeks, 494 students), each representing a submission (possibly non-existent) by one student. Table 1 describes the features used in this initial clustering.

Table 1. Features used in initial clustering

Feature	Description
percent_early	Percentage of attempts made three days or more before the due date
percent_normal	Percentage of attempts made that were neither early nor late.
percent_late	Percentage of attempts made on the due date
num_compile_errors	Number of attempts involving compilation errors.
first_mark	Percentage of tests passed on first attempt.
last_mark	Percentage of tests passed on last attempt.
num_attempts	Number of attempts not involving compilation errors.
time_taken	Indicator for the time between the first and last submission. 0: student only made 1 submission (time between the first and last submission not relevant); 0.5: student took less than 26.45 minutes to complete their task; 1: student took more than 26.45 minutes to complete their task; -100: student did not attempt the task; (forces students who did not submit into their own cluster)
single_attempt	Specifies whether the student made no attempts ("none"), a single attempt ("yes" or multiple attempts ("no").

We note that the features, percent_early, percent_normal and percent_late are dependent. However, removing one would lead to different results depending on which feature was removed, so all were included to preserve symmetry.

We then clustered these 5434 vectors (with k-means algorithm) into six groups, with centroids are summarised in Table 2. Since these clusters would be used to perform further clustering, in which the distance between all clusters would be assumed to be equal, it was important that there were not two similar clusters, or one cluster comprised of what should be two clusters. We experimented with various numbers of clusters in the range of 4-7, and found that 6 clusters best satisfied these criteria.

Table 2. Cluster centroids of submissions clustering

Feature	Full Data	Cluster Number (Number of Vectors)					
		0 (5434)	1 (488)	2 (1017)	3 (903)	4 (719)	5 (607)
% early	0.30	0.55	1.00	0	0.39	0.00	0.00
% normal	0.22	0.19	0.00	0	0.43	0.99	0.01
% late	0.17	0.27	0.00	0	0.18	0.01	0.99
num compile	0.14	0.79	0.08	0	0.06	0.17	0.21
first mark	0.57	0.65	0.96	0	0.68	0.93	0.88
last mark	0.64	0.76	0.98	0	0.96	0.96	0.90
num attempts	0.44	0.59	0.52	0	0.94	0.54	0.52
time taken	-31*	0.78	0.07	-100*	0.74	0.17	0.18
single attempt	yes	no	yes	none	no	yes	yes

The features typical of each of the clusters allow us to interpret the general behaviour captured in these clusters. These are summarised in Table 3 and discussed in more detail below. Note that we refer to the following five grade categories from here on: High Distinction (HD), mark of 85 or above; Distinction (D), mark between 75 and 84; Credit (CR), mark between 65 and 74; Pass (P), mark between 50 and 64; Fail (F), mark below 50.

Table 3. Brief description of submissions clusters

Cluster	Typical Behaviour for the submission
0	Early start, steady improvement from CR to D.
1	Early start, strong first attempt.
2	No submission made
3	Normal start, steady improvement from CR to HD.
4	Normal start, strong first attempt.
5	Late start, strong first attempt.

Cluster 0: Attempts in this cluster were started early and progressed for a long and had a high number of compile errors in the attempts. They contained a medium number of attempts, and their improvement was moderate: attempts began with around a credit and improved to a distinction. (9% of vectors were in this cluster).

Clusters 1, 4 and 5: these represent cases where students performed well in the weekly task and began early, neither early nor late, and late respectively. Students, when in any of these three clusters, on average began with an initial and final mark of HD. However, Cluster 1 students had the highest average mark in both cases (96-98), followed by Cluster 4 (92-96), then Cluster 5 (88-90). These students usually made a medium number of attempts with a small number of compilation errors over a small amount of time. (19, 13 and 11% of instances respectively).

Cluster 2: This cluster represents cases where students did not attempt the task. (31% of cases).

Cluster 3; The high number of submissions and time taken suggests students, when in this cluster, put in the most effort. Improvement was typically large – from around a low credit (68) to an HD (96). The majority of these students' attempts were not late, and there were a low number of compilation errors. (17% of instances).

Intuitively, we would describe Clusters 0 and 3 as the behaviours that make best use of the autograding system, by making use of the feedback to achieve a significantly higher final grade.

Clusters 1, 4 and 5 are interesting because these behaviours are unlikely to benefit from being able to make multiple attempts, since early attempts are already of a high quality. It might be that students who found a task easy to complete in one week may not feel the need to invest time early in subsequent.

Figure 2 shows the general distribution of behaviours each week. We can see that many students were in Cluster 1 in the first week, probably due to the simplicity of the task, and that the number of students who did not submit at all (Cluster 2) is similar from week 2 to week 8, but increasing towards the end of the semester, especially in weeks 9, 10 and 11. This can be explained by the fact that these weeks are heavy in assignment deadlines in all the courses, including this course.

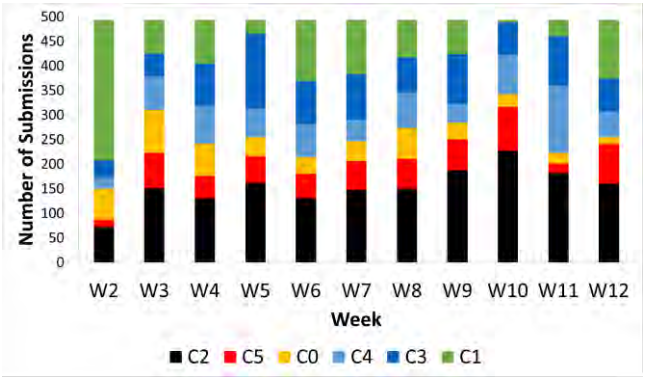


Figure 1. Number of students in each submission cluster each week. Order of clusters follows order discussed in Section 4.4.

4.2 Evolution of students with different exam grades

Figure 3 shows the relationship between the submission clusters each week and the final exam grades of students corresponding to those clusters.

We chose to study the relationship of the submission clusters with the final exam since it is the main and most comprehensive assessment component in the course. It is worth 60% of the final mark, covers all topics and is highly correlated with the final mark for the course. Here we use the same grade categories as previously: HD, D, CR, P, F. NA denotes students who did not sit the exam. There is a minimum requirement policy of scoring at least 40% in the final exam to pass the course: this means that even if students scored very high during the semester (say, 100% of 40), they would fail the course if they scored less than 40% at the final exam (say 30% of 60), even though their raw mark would be above a pass (58%).

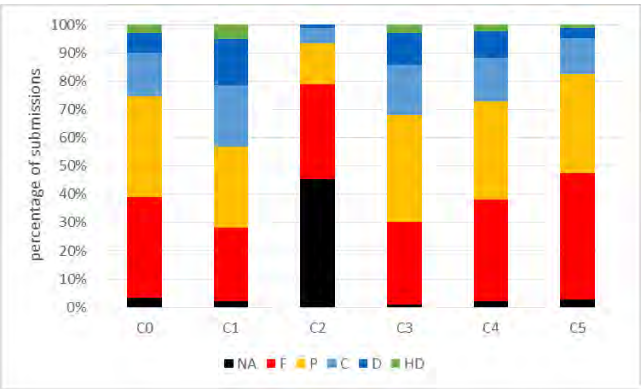


Figure 3. Percentage of submissions in each submission cluster each week with the submitting student's final exam grade

We can see that the students who obtained HD and D in the exam were often in Cluster 1 during the semester and also sometimes in Clusters 4 and 3. These clusters corresponding to the best performing students during the semester, with Cluster 1 containing the students who start early with a very high initial mark, Cluster 4 – the students who start normally with a high mark and Cluster 3 – the students who start early or normally from an average mark and work very hard to improve their submissions.

The students who obtained CR and P at the exam did not show a predominant behavioural pattern during the semester when completing the weekly tasks – they belonged to all clusters. However, more P than CR students were in Cluster 2 (the cluster of students who did not submit), for all weeks. In contrast, very few of the CR students were in Cluster 2 in the early weeks although this number increased after week 8.

A large proportion of the students who failed the exam were in Cluster 2 during the semester, but there are failing students in all behavioural clusters. The students who did not sit the exam are predominantly from Cluster 2 and, from Figure 1, their number is relatively stable from week 2 to week 12, which shows that most likely these students dropped out early in the semester.

4.3 Evolution of students from a given cluster

We can also follow the evolution of the students from a given cluster from a specific week. For example, starting with the six clusters from Week 3, we can analyse each cluster separately and investigate where the students from each cluster go in the subsequent weeks, as shown in Figure 2.

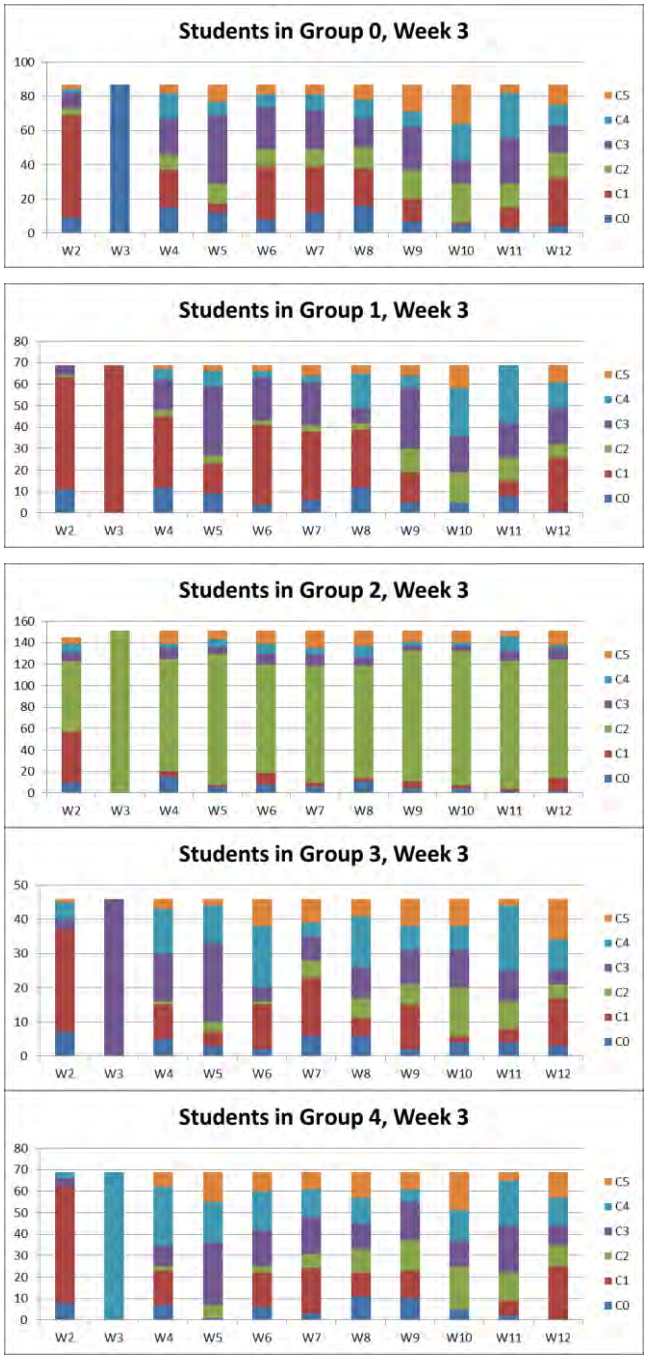


Figure 2. Analysing the six clusters from week 3 separately - percentage of students and each each cluster in subsequent weeks

The graphs show that the students from Cluster 0 in week 3 were mainly in Clusters 1 and 3 in the following weeks, i.e. they were able to achieve a higher mark on the weekly tasks compared to week 3. The students from Cluster 1 in week 3 mainly stayed in the same cluster or moved to Cluster 3, i.e. had to put more effort to maintain high marks. The students from Cluster 2 in week 3 (the non-submitting students) stayed in the same cluster with very few exceptions. The students from Clusters 3 and 4 together stayed in these clusters, and the students from Cluster 5 in week 3 moved

between Clusters 3, 5 and 2 during the semester, i.e. they were not always able to achieve high mark, possible because they started late, and also did not submit in some weeks, e.g. week 10.

We can clearly say that extracting patterns based on visual analysis of the graphs is difficult. This motivated our second clustering of behavioural data described in the next section.

4.4 Comparing the clusters in the middle and end of the semester

To better understand the stability of the clusters over time, we conducted clustering in the middle of the semester (after week 7) using the same method as described in Sec 4.1. We then compared the new clustering to the old clustering, described in Sec 4.1, to determine whether the end-of-semester clusters had already formed midway through the semester. Note that the clustering in both cases is done using all the available data at that time point, i.e. the mid-semester (early) clustering uses the data from week 2 to week 7, and the end-of-semester (end) clustering uses the data from week 2 to week 12.

In both cases, we followed the same clustering procedure – one example represents one submission. We paired each early cluster with a corresponding end cluster, seeking to maximize the overlap between the matched clusters.

More precisely, we considered the bijection, m, from the set of end clusters to the set of early clusters which minimized the distances between the centroids of each late cluster c_i and the paired early cluster $m(c_i)$. We then defined the accuracy of m on an early cluster $m(c_i)$ to be the proportion of submissions in end cluster c_i that were also in early cluster $m(c_i)$. That is,

$$accuracy(m(c_i)) = \frac{|S(m(c_i)) \cap S(c_i)|}{|S(c_i)|}$$

where i is an integer from 0 to 5, $S(x)$ denotes the set of submissions assigned to the cluster x, and $|X|$ denotes the number of elements in set X.

The chosen bijection gives the accuracies shown in Table 4. We can see that the accuracy of the mapping of four of the end clusters (1, 2, 3 and 5) is very high (≥90%). This is to be expected of Cluster 2 as all non-attempts are forced into their own cluster. However, this is not the case for Cluster 1, Cluster 3 and Cluster 5, and the high accuracy indicates that these clusters had already formed midway through the semester. End Cluster 4 had also emerged in week 7, as evident by relatively high accuracy of the mapping to it (76%), but had not stabilized yet. The mapping of end Cluster 0 had a low accuracy, indicating that this cluster had not yet been formed in week 7. A closer examination shows that the students in early Cluster 0 used strategies typical not only of end Cluster 0 but also of end Clusters 1 and 4, as well as end Clusters 5 and 3, to a lesser extent.

Table 4. Accuracy of each cluster in the middle of the semester (week 7) relative to the end of the semester (week 12)

End cluster (week 12)	0	1	2	3	4	5
Accuracy in week 7	13%	90%	100%	91%	76%	97%

In summary, the comparison of the end clusters from week 12 with the early clusters from week 7 shows that most of the end clusters had already formed or emerged in the middle of the semester. We can use these results to provide feedback to students in the middle of the semester and devise appropriate early intervention.

4.5 Behavioural evolution in time

The submission clustering in section 4.1 gave us clusters capturing behaviour per student per weekly task. An interesting question is how each student's behaviour evolved during the semester in regards to their weekly task. In order to explore this question, we performed an additional clustering to identify groups of students with similar submission behaviours over the weeks. The features used for this clustering try and capture the variety and frequency of behaviours (in terms of submission clusters found in 4.1). Note that features, c0-c5 count, are dependent, since the number of weeks are fixed. However, as previously, we maintain all to preserve symmetry. These features are described in Table 5. K-means clustered students into 6 groups, where the number of clusters was determined empirically. The centroids of this new clustering, which we call behavioural clustering, are shown in Table 6.

Table 5. Features used in behavioural clustering

Feature	Description
num_clusters	Number of submission clusters a student's submission belonged to over the semester
c0_count	Number of weeks where a student's submission belonged to behavioural cluster 0
c1_count	Number of weeks where a student's submission belonged to behavioural cluster 1
c2_count	Number of weeks where a student's submission belonged to behavioural cluster 2
c3_count	Number of weeks where a student's submission belonged to behavioural cluster 3
c4_count	Number of weeks where a student's submission belonged to behavioural cluster 4
c5_count	Number of weeks where a student's submission belonged to behavioural cluster 5

Before we describe these clusters, we also examined the relationship between final exam marks and a student's behavioural cluster. Figure 3 shows the percentage of students in each behavioural cluster receiving each of the possible exam grades: HD, D, CR, P, F and NA, where NA indicates that a student did not sit the final exam. The behavioural clusters in this figure have been ordered from lowest to highest based on the percentage of students passing the final exam in those clusters (i.e. behavioural clusters 3, 4, 1, 5, 2, then 0). We see in general that the proportion of passing students that receive higher bands increases, as well as the proportion of students who sit the final exam.

Table 6. Behavioural cluster centroids

Feature	Full Data	Behavioural Cluster Number					
		0	1	2	3	4	5
num_clusters	3.92	4.17	5.13	4.41	1.31	3.48	4.42
s0_count	0.99	1.00	1.51	1.49	0.14	0.66	0.83
s1_count	2.06	5.26	1.44	2.21	0.11	0.93	1.48
s2_count	3.44	0.69	2.12	0.72	10.65	7.20	0.63
s3_count	1.83	2.22	1.61	4.49	0.04	0.52	2.14
s4_count	1.46	1.42	1.43	1.31	0.03	0.41	4.52
s5_count	1.23	0.42	2.90	0.77	0.04	1.28	1.41

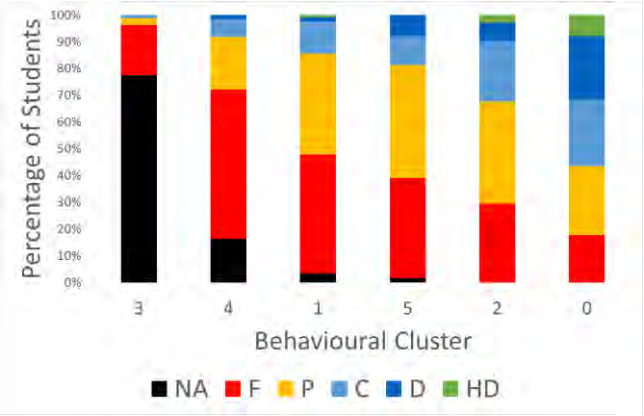


Figure 3. Exam performance of students in behavioural clusters, ordered in increasing proportion of students passing their final exam

We note that over 80% of students in Behavioural Cluster 0, which comprised 20.4% of the cohort, passed the final exam – the highest percentage of all the secondary clusters. In addition, over 50% of students in this behavioural cluster received at least a credit.

Behavioural Cluster 2 had the next highest pass rate of around 70%. The proportion of students receiving high bands in this cluster was lower than Behavioural Cluster 1, but greater than in other clusters.

Using the cluster centroids in Table 6, the weekly behaviours typical of different behavioural clusters are summarised below, in the cluster order used in Figure 3.

Behaviour Cluster 3: These students belonged to an average of 1.3 different clusters throughout the semester. 96.8% of the time they were assigned to Submission Cluster 2, indicating that they almost never completed their weekly tasks. These students may have dropped out of the course during the semester. (16.2% of students).

Behavioural Cluster 4: These students oscillated between an average of 3.5 clusters throughout the semester. 65.4% of the time, they fell into submission Cluster 2, indicating that they frequently did not complete their weekly tasks. However, these students belonged to submission Cluster 5 11.6% of the time, suggesting they sometimes started late but still performed well. From this, we see that these students are possibly quite capable, but do not put much effort into their weekly tasks.

Behavioural Cluster 1: These students were in an average of 5.1 submission clusters over the semester. Cluster 5 was the most common submission cluster, which students were in 26.3% of the time, followed by Cluster 2 (19.3%), Cluster 3 (14.6%), Cluster 0 (13.8%), Cluster 1 (13.1%) and Cluster 4 (13.0%). Thus these students often started late but did well, but also often didn't submit at all. These students sometimes worked hard and achieved high marks, sometimes worked hard without achieving high marks, sometimes began early and did very well and sometimes began neither early nor late and did well. These students displayed inconsistent behaviour over the weeks, sometimes putting in a great amount of effort and sometimes not trying at all. (24% of students).

Behavioural Cluster 5: These students belonged to an average of 4.4 different clusters over the semester. They fell into submission Cluster 4 the most often - around 41.1% of the time – followed by submission Cluster 3 (19.5%), Cluster 1 (13.5%) and Cluster 5 (12.8%). Thus these students very often started their weekly tasks neither early nor late and did well, commonly started early and worked hard until they did well, sometimes started early from a high mark and sometimes started late from a high mark. (13 % of students)

Behavioural Cluster 2: These students belonged to an average of 4.4 different submission clusters over the semester, with Cluster 3 being the most common (40.8%), then Cluster 1 (20.1%), Cluster 0 (13.6%) and Cluster 4 (11.9%). Thus, these students commonly began early with a medium mark, worked hard and achieved good marks. They also often started early from a high mark, sometimes worked hard without achieving a high mark and sometimes started neither late nor early with a high mark. These are hard-working students who often found the tasks challenging, but still did fairly well in them.

Behavioural Cluster 0: Finally, in the behavioural cluster with the highest final exam pass rate, students oscillated between an average of around 4.2 clusters in the course of the semester. They were in submission Cluster 1 47.8% of the time, Cluster 3 20.2% of the time, Cluster 4 12.9% of the time and cluster 0 9.1% of the time. This suggests these students started early with high marks around half the time. They often started early with medium marks, but worked hard until they achieved a high mark and sometimes started neither late nor early, achieving high marks. Occasionally they worked hard without achieving high marks. (20% of students). These students often did well on their first submission but, when they didn't, they worked hard to achieve high marks.

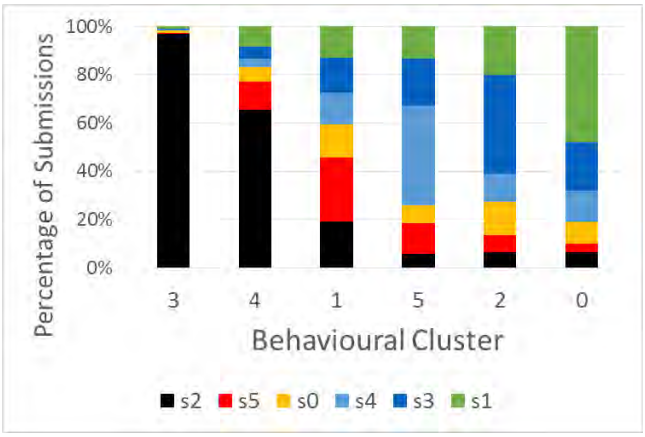


Figure 4. For each behavioural cluster, the percentage submissions in each submission cluster (s0, s1, s2, s3, s4, s5)

4.5.1 General Trends

By analysing behavioural clusters and the most common submission clusters the students' submissions were in, we noticed general trends as the final exam pass rate increased. For example, submissions in Submission Cluster 2, characterised by no submission attempt, were most common in students in behavioural clusters with the lowest pass rate. On the other hand, Submission Cluster 1 (early start, strong first attempt) was most common in behavioural clusters with higher pass rates. We used these trends to order the submission clusters: Submission Clusters 2 and 5, being the most and second most common submission clusters in poorly performing behavioural clusters, were placed on the bottom of the scale. Of the remaining four submission clusters, Submission Cluster 0 was least common in the top three behavioural clusters,

and so came next on the scale. This was followed by Submission Clusters 4, 3 and then 1, which became more prevalent in higher performing behavioural clusters. Figure 4 shows the percentage of submissions in each behavioural cluster that fell into each submission cluster. The behavioural clusters are ordered based on pass rate, and the submission clusters are ordered as described above. The prevalence of each submission cluster in different behavioural clusters is summarised in Table 7.

Table 7. Submission Clusters Typical of each Behavioural Cluster

Submission Cluster	Common in Behavioural Clusters with	Submission Cluster Description
0	Many different pass rates	Average students, medium/high effort.
1	High pass rates	Excellent students who started early from a very high mark.
2	Low pass rates	Did not submit.
3	High pass rates	Hard working students – from CR to HD.
4	Medium pass rates	Good students who started neither early nor late from a mid HD.
5	Low pass rates	Good students who started late from a low HD and improved slightly.

4.5.2 The median

We can also visualise the evolution of student behaviour over the semester in a meaningful way. We looked at the weekly behaviour of students in each behavioural cluster each week and found the "median" behaviour. This was achieved by taking the median of each original feature for these students, such as the first mark, last mark, time taken and percentage of early submissions. We then used this to create a median vector, and found which submission cluster the vector belonged to. We repeated this for all behavioural clusters and plotted the results. This can be seen in Figure 5. Note that submission clusters were previously ordered so the higher the submission cluster the more typical it is overall of the behavioural clusters with the highest pass rate.

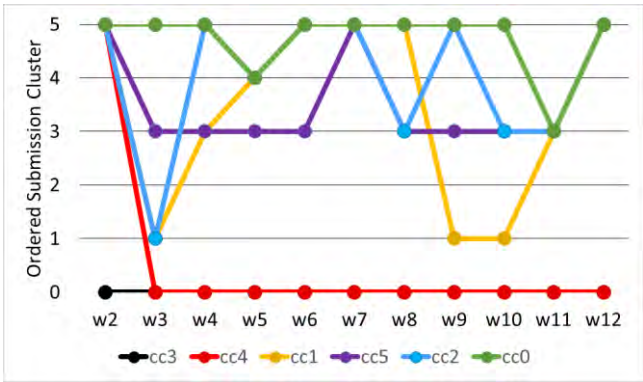


Figure 5. Changing student behaviour over the semester. Each colour represents a behavioural cluster. The median behaviour of students each week (i.e. the median submission cluster) is shown. The submission clusters are ordered so that higher corresponds to better performance.

Rather than the secondary clusters slowly diverging over time, we notice a clear separation from as early as week 3. The secondary clusters with the lowest (secondary clusters 3 and 4) and highest (secondary cluster 0) pass rates are already distinguishable from the other clusters at this time. This early separation of behaviours could facilitate early identification of students at risk of failing or performing poorly, allowing for intervention.

5. DISCUSSION

The scheme in our analysis can be separated into two parts:

(i) A submission clustering, where the approach and performance of each student in each weekly submission is treated as independent and then clustered to give typical task-level behaviours.

(ii) A behavioural clustering, where students are clustered based on the submission clusters they were in over the entire semester.

Through the example of a junior computer science course, we demonstrated the usefulness of this double-clustering method in allowing us to identify some important approaches students in this course took to their weekly tasks. We found that many students started sufficiently early and invested time to improve their attempts based upon instant feedback they received from the autograding system, benefiting from a significant improvement in the quality of their final attempts (Clusters 0 and 3, 26%). We also found that students often found the task sufficiently easy and that further improvements were of little value (Clusters 1, 4 and 5, totaling 43%), and that it was also common for students to not attempt the tasks at all (Cluster 2, 31%). A broader application of this analysis over multiple units of study and across multiple offerings of the same course would be useful in understanding how common such behaviours are in general as opposed to this specific offering.

Through the behavioural clustering, we were able to identify common behavioural patterns over the entire semester, and to draw links between these patterns and final exam outcomes. In particular, we identified behavioural patterns associated with high and low final exam grades. For example, students in behavioural clusters with high pass rates tended to consistently start early with a high mark, or start early and work hard until a high mark was achieved. Conversely, students in behavioural clusters with low pass rates often did not submit their tasks at all. Knowledge of the relationship between behavioural patterns and exam performance is essential in the identification of students at risk of performing poorly and important in the structuring of a course to maximise student learning and performance.

We compared submission clusters that used all data up to week 12 with submission clusters that used all data up to week 7, and found that they were quite similar. This suggests that the typical task-level behaviours of students did not vary much at the end of the semester and that, as a consequence, these behaviours could be identified early on in the semester. Moreover, we saw that the term-long behavioural clusters we found did not slowly diverge over time, but rather there was an immediate difference from as early as week 3. This suggests that both the submission and behavioural clustering could be performed early in the semester, with potentially similar results to the end of semester, allowing for early identification of students at risk of performing poorly and early intervention. We suggest an avenue of future research could be to apply this technique midway through the semester and evaluate its

effectiveness in facilitating interventions that could improve student outcomes.

We also suggest investigating how effective this method can be in general, by applying it to courses with different assessment structures and content, and also to compare the results obtained through these clustering methods to traditional measures of behaviour and engagement, such as tutorial attendance and feedback surveys, to evaluate how well they corroborate.

Although the reported analysis is for data from a system for assessing computer code submissions, it could just as readily applied to other systems in which students can make multiple submissions in response to feedback. For instance, many Learning Management Systems provide multiple-choice style questions for which students can receive feedback about their choices, and this style of question could be used in any discipline. Our analysis depends only upon records of the time and quality of each submission. While we include details such as number of compile errors as one measure of quality, this could readily be substituted with other measures.

6. CONCLUSION

In this paper we have presented a method for analysing student behaviour and the evolution of this behaviour over the semester, using data from autograding system logs. We have shown that this method can be useful in identifying common weekly behaviours of students, and following the changes of such behaviours over the semester. We have discussed the relationship between these behaviours and final exam results, and demonstrated how these behaviours might be detectable early enough in the semester for instructors to intervene. As such, we believe that the techniques discussed here may be implemented and improved upon to realise the full potential of increasingly common autograding systems in facilitating real improvement in student outcomes.

7. ACKNOWLEDGMENTS

This work was funded by the Human-Centred Technology Cluster of the University of Sydney.

8. REFERENCES

[1] Sherman, M., Bassil, S., Lipman, D., Tuck, N. and Martin, F. 2013. Impact of autograding on an introductory computing course. *Journal of Computing Sciences in Colleges*, 28 (6), 69-75.

[2] Enstrom, E., Kreitz, G., Niemela, F., Soderman, P., and Kann, V. 2011. Five years with Kattis - using an automated assessment system in teaching. *In Proceedings of the Frontiers in Education Conference (FIE)*, IEEE.

[3] Milojicic, D. 2011. Autograding in the cloud: interview with David O'Hallaron. I*EEE Internet Computing* 15 (1), 9-12.

[4] Gramoli, V., Charleston, M., Jeffries, B., Koprinska, I., McCrane, M., Radu, A., Viglas, A., and Yacef, K. 2016. Mining autograding data in computer science education. In *Proceedings of the Australasian Computing Education Conference (ACE)*.

[5] Koprinska, I., Stretton, J., and Yacef, K. 2015. Predicting student performance from multiple data sources. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*, LNCS 9112, 678-681.

[6] Koprinska, I, Stretton, J., and Yacef, K. 2015. Students at risk: detection and remediation. In *Proceedings of the International Conference on Educational Data Mining (EDM)*, 512-515.

[7] Perera, D., Kay, J., Koprinska, I., Yacef, K., and Zaiane, O. 2009. IEEE Transactions on Knowledge and Data Engineering, 21(6), 759-772.

[8] Tomkins, S., Ramesh, A., and Getoor, L. 2016. Predicting post-test performance from online student behaviour: a high school MOOC case study, In Proceedings of the International Conference on Educational Data Mining (EDM).

[9] Radu, A. and Stretton, J. PASTA, School of Information Technologies, University of Sydney, http://www.it.usyd.edu.au/~bjef8061/pasta/. Accessed: 2016-05-02

[10] INFO1105: Data Structures (2015 - Semester 2). https://cusp.sydney.edu.au/students/view-unit-page/uos_id/79883/vid/309891. Accessed: 2016- 03- 05.

Modelling the way: Using action sequence archetypes to differentiate learning pathways from learning outcomes

Kelvin H R Ng
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
e140025@e.ntu.edu.sg

Kevin Hartman
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
khartman@ntu.edu.sg

Kai Liu
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
kliu006@e.ntu.edu.sg

Andy W H Khong
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
andykhong@ntu.edu.sg

ABSTRACT

During the semester break, 36 second-grade students accessed a set of resources and completed a series of online math activities focused on the application of the model method for arithmetic in two contexts 1) addition/subtraction and 2) multiplication/ division. The learning environment first modeled and then supported the use of a scripted series of steps for solving mathematical word problems. As students completed the activities, the learning environment captured their event-related data. We then used a combination of Affinity Propagation, an automated form of clustering, and sequential pattern mining to convert the activity logs into interpretable activity sequences. Analysis of the activity sequences identified distinct patterns of behavior that strongly predicted which students would transit from the familiar addition/subtraction word problem activity to the unfamiliar multiplication/division word problem activity. Students who showed the greatest and least compliance with the script were the least likely to attempt the multiplication/division activity. Students who showed more of a schematic problem solving process were more likely to continue to the multiplication/division activity.

Keywords

Sequential pattern mining, affinity propagation, cognitive models

1. INTRODUCTION

1.1 Mathematics Learning via the Model Method

In Singapore, early-elementary students are taught to solve arithmetic word problem via the model method [1]. This systematic approach is based on Polya's problem solving techniques [2]. The method can be broken into five steps known as the RIGHT sequence. When applying the RIGHT sequence, students 1) read the word problem, 2) identify the nouns, numeric values, and unknown variable to be solved, 3) graph these values in a box diagram, 4) indirectly perform the appropriate calculation by reasoning through the diagram, and 5) review their work.

The RIGHT sequence, as a learning mnemonic, provides students with a script for executing the model method. Scripts are collections of discrete actions that, when followed, achieve a goal or specific outcome [3]. Ordering food at a restaurant serves as the classic example of following a cognitive script [3]. In most dining establishments, the same set of steps, with some allowance for minor deviations, will lead the patron to receive a meal. Similarly, following the RIGHT sequence will lead students to the correct answer to a word problem. Scripts have been found to reduce cognitive load for novice learners by lessening the mental resources needed for planning and completing the plan. Scripts also lead to greater expressions of automaticity by experts [4]. However, cognitive psychologists also view scripts as the most nascent form of schemas [5]. The application of scripts is contextually bound and rather inflexible. Schank and Abelson [6] refers to scripts as event schemas which are task specific and order dependent. The previous restaurant script may work for purchasing food at most dining establishments, but it could not be used successfully to purchase food at a supermarket. To negotiate the supermarket, one would need to apply either a different script or rely on a more generalizable schema.

Generalizable schemas consolidate the steps of an event schema under a larger label [7]. Rather than simply ordering a meal at a restaurant, a generalizable schema for acquiring food would include all of the known methods of gaining nourishment. What generalizable schemas sacrifice in terms of automaticity, they make up with flexibility [5].

Returning to the original example of the model method, the intent behind introducing students to using box diagrams to solve algebraic word problems is to give them a generalizable schema for solving real-world problems [1]. In practice, students often instantiate the schema in the form of a word problem solving script [8]. When looking at problem solving accuracy, teachers cannot diagnose whether a student has internalized the model method as a generalizable schema or as a problem solving script because both strategies work in the short term. However, only the generalizable schema prepares students to flexibly transfer the model approach to new situations. In this study, we sought to generate an algorithm to classify students as exhibiting script-like or generalizable schema-like behaviors in the context of a series of online math enrichment activities. We then tested whether script-like behaviors, generalizable schema-like behaviors, or problem solving accuracies were more predictive of students seizing future learning opportunities.

1.2 Machine Learning and Temporal Sequencing

In the context of this paper, we define an action as a single line item in a log file and action sequences as the collection of actions that can be described with a more general semantic label. For instance, entering a number into a text box constitutes an action. All of the various combinations of actions that lead to the calculation of that

number being entered into the text box constitute a single action sequence.

When attempting to identify meaningful action sequences while preserving the temporal relationships between those actions, educational data miners use techniques like process mining and sequential pattern mining. With process mining, the learning pathways students take within a learning environment are identified and visualized [9, 10]. Deviations in these pathways from the intended pathways can then be analyzed for meaning [9, 10]. Alternatively, sequential pattern mining identifies frequently occurring subsequences within a temporal dataset for further analysis. Recently, Ye et at. used a hierarchical variant of SPAM to analyze data collected from Betty's Brain OELE [11]. The analysis illustrates the importance of using temporal relationships between user activities to make predictions about future learning behaviors [11]. Veeramachaneni, Adl, and O'Reilly [12, 13] also highlight the significance of incorporating a range of temporal dependencies into features when predicting student traits. Applying a crowd sourcing technique, they obtained lists of complex features that, when divided, seem obvious to experienced teachers and data scientists. However, neither group could have generated the entire list of the features on its own [12].

When extracting frequent patterns from unstructured data, sometimes the patterns are composed of short sets of actions which actually belong to longer action sequences. These algorithms have a tendency to obscure the temporal relationships between the extracted features. Additionally, sequencing combinations of actions and filtering out rare patterns rather than using the complete action sequences can result in the loss of rare action combinations that achieve a common action sequence [14]. The potential for losing rare actions belonging to common action sequences is magnified in learning environments populated by novice learners. Novice learners who are introduced to a learning environment have the dual task of learning to navigate the environment as well as gaining competency with the concepts central to the learning activities. In such situations, data mining techniques that analyze learner actions more schematically, rather than in scripted terms, may actually yield more parsimonious models.

With the goal of aligning our data mining techniques with learners' mental schemas, we propose conceptually reframing individual actions as words and action sequences as sentences. With this recasting, we can apply a combination of string distance measures that take into account the vocabulary and word order within the sentences to make pair-wise comparisons. We used an Affinity Propagation (AP) [15] algorithm to recover distinct action sequences that translated to learning behaviors and the sequence exemplars are referred to as action sequences archetypes (ASAs). Sequential pattern mining is applied to cluster members to summarize the temporal deviations within each cluster. The described method preprocesses the data for analysis and interventions to steer learners towards desired educational outcomes.

AP is useful for our particular context because it simultaneously considers all data points in relation to a shared preference to determine a suitable number of output clusters. This structure independence lends AP to situations where there is no a priori expectation about the output cluster size or number [15]. In our case, the number of sequences within the dataset varies greatly between sessions. Beyond accommodating this variability, the algorithm's input, a similarity matrix defined by the pairwise similarities between two sequences, is not limited to symmetrical pairwise similarities. This freedom creates opportunities to differentiate the discrete ordered lists using different distance measurements. We augmented the AP algorithm with a tree-based sequential pattern mining algorithm for its ability to handle multiple minimum supports and rare item filtering [14]. The algorithm is used to extract maximal sequences, which are longest sequences that satisfy the minimum frequency threshold, for each cluster.

2. Data Collection

36 second grade students completed the first phase of activities in the online learning environment during the school holidays. The activities were part of an ``out of school" enrichment opportunity. At the onset of data collection, all of the invited participants had previously received formal instruction from their teachers on using the model method to solve addition and subtraction word problems. The students had not yet received instruction within the school curriculum on using the model method with multiplication and division word problems.

The online learning environment offers two phases of content. During Phase 1, students' complete addition and subtraction activities. In Phase 2, students encounter multiplication and division activities. Each content phase is divided into four sets of activities: 1) video tutorials, 2) structured activities, 3) unstructured activities, and 4) multiple choice questions (MCQ). The video tutorials explain the RIGHT sequence and the use of the model method in a pen-and-paper context. After each video, students receive a set of practice exercises related to the content of the video tutorial. Additional video segments at the start of each practice question introduce the recommended sequence of steps to solve the word problems using the model method and the representational supports found within the learning environment. The representational supports include using the highlighted noun blocks and the RIGHT checklist while answering the word problems.

The structured activity focuses on the "G" in the RIGHT sequence. Each question in the activity is presented with a practice word problem. The problem is displayed with four multiple choice options showing different bar diagrams and a checklist in the right corner of the workspace. The checklist shows the first three steps of the RIGHT sequence. Students are advised to tick off the respective check boxes as they complete each step in the RIGHT sequence. In the structured activity, the checklist is limited to the first three steps of the RIGHT sequence as students are not expected to take their model to completion.

After students identify the model they think matches the content of the word problem, they are given feedback about their choice before moving on to the next question. They are presented with options to review, ask for hints or proceed to the next question. Choosing to review the question returns students to the last snapshot of the question before the answer submission. Requesting a hint provides students with a partially completed model as a guide. Hints are given progressively until the complete model is revealed. Two hints can be requested for each question. If a student chooses to proceed to the next question without reviewing errors after submitting an error, the learning environment logs the action as ignoring an error.

In the unstructured activity, students solve the problems using the RIGHT sequence. A snapshot of the learning environment for this activity prior to any attempt is shown in Figure 1. Model templates for all four arithmetic operations are made available for students to complete with the correct numerical values. Nouns mentioned in the problem are also presented as colored blocks for labeling the relevant model. Students can drag and drop the blocks to their

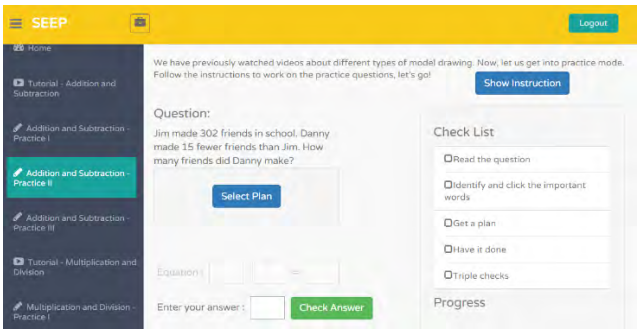


Figure 1: Workspace for unstructured activity.

selected model. Students may also enter mathematical expressions in the provided text boxes. Alternatively, students may forego performing any or all of these actions. However, they must submit a final answer before receiving feedback about their answer and proceeding to the next question.

For the MCQs, students are presented with a page containing ten multiple-choice questions. Each question requires inputting a numerical answer into a textbox. Students again have the option of using the RIGHT checklist that floats in the right margin of the screen. The checklist resets whenever a student interacts with a different question. Students must complete all of the activities before proceeding to Phase 2.

3. Data Preprocessing

Only clickstream and navigation information occurring within the online module was recorded to the log file as students worked through the activities. Beyond navigation and interface information like mouse clicks and text entries, off-task behavior like leaving the learning environment by activating another browser tab and returning to the online module was also collected. A total of 23233 log entries were collected. Table 1 lists the recorded actions.

The log entries were preprocessed to indicate the use of the different learning resources within the learning environment. For example, highlighting a keyword within a question is recorded as one log entry per keyword. However, only the first instances of highlighting and canceling of highlights are retained for each question attempt to signal that the highlighting resource was used. In addition, while learners navigate through the model template selection, we only analyze the final template selection instead of considering all of the navigation activity within the selection area. Filtering out these events greatly reduces the amount of variability within the action sequences and makes them more schematic. To identify revision of answers, first selections for the MCQs are labelled as *mcq_select.* Additional selections are labelled as *mcq_alter*. Following the described procedure reduced the size of the dataset to 9918 entries, or 275 entries per student. The maximum number of analyzed actions for a student was 868. The final list of actions for each type of activity is shown in Table 1.

In the reduced dataset, each action sequence is identified and labelled. For videos, an action sequence constitutes the actions taken from the start of a video to terminating the video either by completing the video or navigating away from the current page. For the exercises, the action sequences span from the initiation of a question until the user proceeds to the next question.

Table 1: List of all log actions

Action	Video	Structured	Unstructured	MCQ
leave_page	✓	✓	✓	✓
return_to_page	✓	✓	✓	✓
phase_start	✓	✓	✓	✓
phase_stop	✓	✓	✓	✓
video_start	✓			
video_stop	✓			
video_pause	✓			
video_scrub_foward[1]	✓			
video_scrub_back[1]	✓			
video_end[1]	✓			
video_end_full[1]	✓			
video_replay	✓			
video_select_same	✓			
video_select_diff	✓			
attempt_qn		✓	✓	✓
highlight		✓	✓	✓
undo_highlight		✓	✓	✓
check_checklist		✓	✓	✓
mcq_select[2]		✓		✓
mcq_alter[2]		✓		✓
confirm_model[2]			✓	
mouse_drag[2]			✓	
label_model[2]			✓	
label_eq			✓	
submit[2]		✓	✓	✓
review_error		✓	✓	✓
ignore_error		✓	✓	✓
show_hint		✓	✓	✓

[1] Actions are inferred from clickstream data due to limitation of YouTube's application programming interface (API).

[2] Actions are recorded but filtered out for the purpose of this analysis.

4. Techniques

4.1 Distance Measures

To differentiate action sequences as one would differentiate sentences, it is necessary to consider the vocabulary (actions) of each action sequence and the order of those words. Our proposed distance measure includes four components, a modified version of the common word order measure [16], Jaccard distance, length difference, and vocabulary rarity. The features capture different aspects of action sequences for differentiation. The distance measure between two action sequences S_1 and S_2 is given by the weighted sum of all four features. In this paper, a constant weight is assigned across the four features.

$$\begin{aligned} dist(S_1, S_2) = {} & w_1 * JaccardDist(S_1, S_2) \\ & + w_2 * CWO(S_1, S_2) \\ & + w_3 * \max\left(idf_{t_j \notin S_1 \cap S_2}(t_j, D)\right) \\ & + w_4 * abs\big(length(S_1) - length(S_2)\big) \end{aligned} \quad (1)$$

where

$$w_1 = w_2 = w_3 = w_4 = 1 \quad (2)$$

Jaccard distance defined by

$$JaccardDist(S_1, S_2) = 1 - JaccardSim(S_1, S_2) \quad (3)$$

where

$$\text{JaccardSim}(\text{S}_1, \text{S}_2) = \frac{|\text{S}_1 \cap \text{S}_2|}{|\text{S}_1 \cup \text{S}_2|} \quad (4)$$

captures the degree of dissimilarity between two sequences through the number of unique terms that are not common to both. The Jaccard distances are derived from Jaccard similarity which determines the ratio of unique common actions between two action sequences. Jaccard similarity and distances are bounded between zero and one.

In our context, the common word order measure reflects the similarity of the order in which actions appear between two action sequences. The measure equals zero when the common actions of two sequences occur in the same order and reaches a maximum of one when the common actions appear in reverse order. Given two sequences $A = \{a_1, a_2, \dots, a_n\}$ and $B = \{b_1, b_2, \dots, b_m\}$ composed of l common action, where $l \leq n \leq m$. Retaining only the common actions, sentence $A = \{a_1, a_2, .., a_l\}$ is transformed into a numerical representation $X = \{1,2, \dots, l\}$ by substituting the actions with its indices. The same actions in B are replaced with the same numerical indices to form B. The common word order measure can then be computed by

$$\text{CWO}(\text{S}_1, \text{S}_2) = \begin{cases} 1 - \dfrac{\left(2\sum_{i=1}^{l}|x_i - y_u|\right)}{l^2}, & if\ l\ is\ even \\ 1 - \dfrac{\left(2\sum_{i=1}^{l}|x_i - y_u|\right)}{l^2 - 1}, & if\ l\ is\ odd \\ 1, & if\ l\ is\ odd\ and\ l = 1 \end{cases} \quad (5)$$

The common word order measure is designed for sentences where a bag-of-words representation has a large number of words, most of which have low frequencies. Due to the constraints of the learning environment, our data set contained many actions with high frequencies. Retaining the common terms within action sequences may result in substrings of inequivalent lengths. Therefore, there may exist more than one combination of mapping between these sentences. To remedy this possibility, we adapted the concept of a common word order measure to obtain a distance estimate for the action sequences by first filtering the reduced sequences to remove actions occurring at a specific position that do not contribute to the distance metric. We then match the remaining actions based on their position within the reduced sequence.

The vocabulary rarity is defined as the maximum of the inverse document frequency (idf) [17] of terms that are not common to both sentences. This measure allows us to distinguish sequences that have actions that are less likely to occur from sequences involving trivial navigational patterns. The inverse document frequency of each term t_i in a set of documents D is computed by the logarithmic inverse of the ratio of document counts containing t_i to the total number of documents in the document set D.

$$idf(t_i, D) = \log \frac{|D|}{\{|d \in D : t_i \in d|\}} \quad (6)$$

4.2 Affinity Propagation

The AP algorithm [15] is a message passing clustering algorithm used in image recognition, text comparison and gene clustering. Unlike centroid-based clustering like k-means clustering, AP does not require users to pre-specify the number of clusters and it is less sensitive to parameter initialization [15]. The algorithm takes a pair-wise similarity matrix and a set of shared preferences as inputs to determine the suitability of data points as cluster centroids. Without prior knowledge of the centroids, shared preferences may be set uniformly across all items. When shared preferences are assigned to the minimum value of the pairwise similarity, the number of resulting clusters will also be at its lowest. The inverse is also true. The number of clusters generated by the different shared preference values for the structured activity are shown in Figure 2.

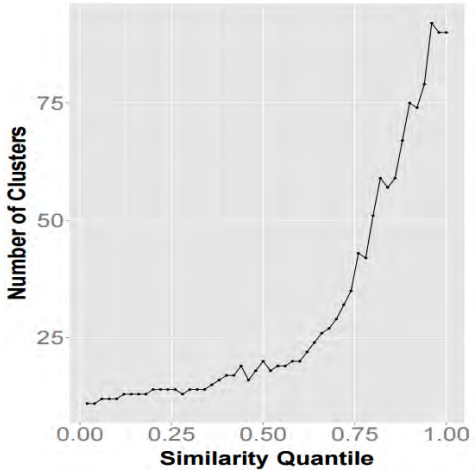


Figure 2: Number of generated clusters based on shared preferences for structured activity.

For our purposes, clusters are determined by passing messages between data points (action sequences) to simultaneously determine their suitability as cluster centroids. The provided similarity matrix may contain unknown pair-wise similarities. However, messages are passed only between points with known similarities. There are two types of messages passed between data points -- responsibility and availability. Responsibility $r(i,k)$, sent from data point i to data point k, dictates the amount of evidence that k is suitable to serve as the exemplar for i, while availability $a(i,k)$, sent from k to i, determines the appropriateness for point i to choose point k as its exemplar. Availabilities are initialized as zeroes and the messages are updated iteratively using

$$r(i,k) = s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\}, \quad (7)$$

$$a(i,k) = \min\left\{0, r(k,k) + \sum_{i' \notin \{i,k\}} max\{0, r(i',k)\}\right\} \quad (8)$$

$$a(k,k) = \sum_{\{i' \neq k\}} \max\{0, r(i',k)\} \quad (9)$$

At the end of each iteration, exemplars are determined from

$$exemplar(i,k) = \underset{k}{\text{argmax}}\{a(i,k) + r(i,k)\} \quad (10)$$

Pairs (i,k) identified from equation (10) state that either data point i will serve as an exemplar for data point k or vice versa. The algorithm terminates only when either a predefined number of iterations is completed or the changes in the messages falls below a certain threshold.

Essentially, the AP algorithm seeks to identify action sequence archetypes (ASA) around which to cluster the remaining action

sequences. After identifying the ASAs, the similar cluster sequences inherit the index of their closest archetype.

4.3 Sequential Pattern Mining

The position coded pre-order linked web access pattern tree mining (PLWAP) algorithm with multiple minimum supports (MMS) [14] is a tree-based sequential pattern mining algorithm. A PLMS-tree is constructed from the logs by adding actions for each learning opportunity sequentially. Each node holds four variables, the label, the frequency count, a binary position code, and a minimum multiple item support (*minMIS*).

The binary code is similar to Huffman coding as it uniquely identifies nodes and subtrees. The root node of the tree is labelled as 0. The leftmost child of any node has a position code of 1 appended to the back of the position code of the node. The position codes of other children are derived from the position codes of their nearest sibling to the left by appending a 0 to the position code.

The support determines the lower bound for frequencies that sequences must satisfy to qualify as a frequent pattern. For multiple minimum support, a minimum support is computed for each unique item in the dataset. In the case of our action sequences, the items in sequential pattern mining correspond to actions in the action sequences. The global minimum support is dictated by the smallest of the minimum supports. Each node maintains its *minMIS* which defines the support required by itself and the suffix tree to qualify as frequent.

As the nodes are added to the tree, a header table is maintained. The header table contains the unique node labels with a list of corresponding binary code of nodes for the same label within the tree. The table is then sorted by order of decreasing frequencies. An example of the PLMS-tree and its corresponding header table is shown in Table 2 and Figure 3.

Table 2: Header table example for Figure 3

Label	Support	Position Code
video_start	10	{01}
video_end_full	6	{011}, {0110001}
video_end	4	{0110}, {011001}, {011000101}
video_pause	2	{011000}
video_scrub_back	1	{01100010}
video_scrub_forward	1	{01100}

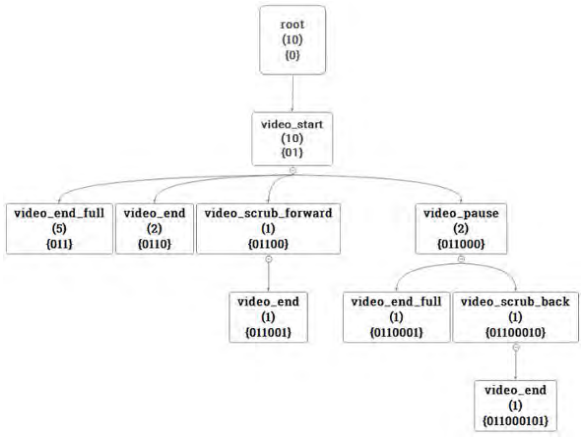


Figure 3: Example of a PLMS-tree for the video activity.

Once the tree is populated, it can be traversed to mine the sequential patterns in the dataset. The mining algorithm proceeds as follows:

1. For each of the entries in the header table, the nodes are identified from the tree using the position codes and the total occurrences is consolidated from the counts of individual nodes. A k-sequence is an ordered list of k items.
   a. If this sequence satisfies its *minMIS*, it qualifies as a 1-sequence.
   b. If the frequency of this node satisfies the global minimum but not its *minMIS*, the label qualifies as a 1-sequence candidate. Candidates are kept as candidates for mining because a subsequent item of lower *minMIS* may qualify these sequences as frequent sequences.
2. The algorithm proceeds to identify the next item in the sequence by scanning the header table.
3. Position codes in the header table containing the position code of the last found node as its prefixes are identified as descendants for that node.
4. The frequencies of the newly identified nodes are aggregated and a new *minMIS* is updated to be the lower of the *minMIS* from previous nodes and the identified node.
5. The algorithm proceeds to search for possible extensions and validates the frequency of these sequences against the *minMIS*.
6. The algorithm terminates when no more descendants are identified from the header table or if the frequencies of the newly identified nodes are less than the value of the global minimum support.

Table 3: Action Sequence Profiles

Activity	No. of Attempts	No. of Profiles
Video	89	10
Structured	286	11
Unstructured	303	11
MCQ	33	12

5. Clustering

Sequences of attempts for each of the four activities are clustered with the AP algorithm. The shared preferences of AP are set to the maximum of the similarity matrices. We use the R package AP for this analysis. PLWAP is then used to retrieve a descriptive summary for each cluster. We restrict the algorithm to only identify contiguous sequences.

We manually merge the clusters into ASAs based on their compositions. The compositions are determined by indicators signaling the use of certain actions between defined checkpoints, similar to the process mentioned in [10]. During the merging process for each archetype, we consider the actions spanning from the onset of a question to the first submission of the question attempt. Descriptions of the ASAs identified in the video, structured and unstructured activities are presented in Table 4, Table 5, and Table 6 respectively.

Table 4: Action sequence archetypes for the video activity

ASA	Description
V1	Offtask
V2	Pre-mature termination
V3	Complete video without other actions
V4	Complete video with pauses
V5	Complete video with off-task
V6	Complete video with pauses and off-task
V7	Complete video with pauses and scrub back
V8	Incomplete video
V9	Incomplete video with pauses
V10	Incomplete video with scrub forward

Table 5: Action sequence archetypes for the structured activity

ASA	Description
S1	Pre-mature termination
S2	Direct answer with off-task
S3	Direct answer
S4	Direct answer with alter of choice
S5	Answer with highlights
S6	Answer with highlights and alter of choice
S7	Answer with checklist
S8	Answer with checklist and highlights
S9	Answer with highlights and alter of choice and checklist
S10	Submission with checklist and highlights but no answer
S11	Submission without answer

Because the MCQ activity presents multiple word problems on the same page, students may freely switch between the problems without signaling their intent. This freedom of choice yields ASAs with indefinite boundaries. The nebulousness of the ASAs associated with each question provides little inferential utility and will not be addressed in the following section beyond attempting to use each student's MCQ accuracy to predict the probability of persisting to Phase 2.

6. Results

6.1 Score-based Prediction of Persistence

We calculated the percentage of questions students correctly answered for the structured, unstructured, and MCQ activities. As shown in Table 7, only a student's MCQ performance is associated with persisting into Phase 2. Knowing students' performances for the structured and unstructured activities leads to a prediction accuracy level similar to that of assuming no student persists from Phase 1 to Phase 2.

As a caveat, the deterministic appearance of the association between MCQ performance and persisting to Phase 2 is misleading. The high correlation is due to the MCQ activity being a prerequisite for Phase 2. The mere presence of an MCQ submission, rather than the score itself, is predictive of persisting to Phase 2. Students who do not make an MCQ submission effectively earn a score of zero for the activity and do not have the possibility to continue to Phase 2. Additionally, all students who do persist to Phase 2 must have scored above a zero on the MCQ activity.

Table 6: Action sequence archetypes for the unstructured activity

ASA	Description
U1	Attempts with no submission
U2	Attempts with off-task and no submission
U3	Submission without attempts
U4	Submission without answer
U5	Submission with answer and highlights
U6	Submission without highlights and drags
U7	Submission without highlights
U8	Submission without drags
U9	Submission without drags with one change of model template
U10	Submission without drags with multiple change of model template
U11	Submission without drags with off-task
U12	Suggested steps
U13	Suggested steps with one change of model template
U14	Suggested steps with multiple change of model template
U15	Suggested steps with off-task

Table 7: Mean activity scores for students who stop during Phase 1 and persist to Phase 2

	Accuracy	
Activity	*Stop-out*	*Persist*
Structured	53.08%	55.23%
Unstructured	54.56%	69.81%
MCQ	0%	91.84%

6.2 Sequence-based Prediction of Persistence

We converted the frequency of each ASA into a percentage of a student's total action sequences. We then used a classification and regression tree (CART) algorithm to predict which students continued on to Phase 2 based on their ASA values. The decision trees associated with progressing based on ASAs from the video, structured and unstructured activities are presented in Table 4, Table 5 and Table 6 respectively.

While persistence cannot be reliably predicted based on video ASAs, it can be accurately predicted by the structured and unstructured ASAs. The predictability of these features is

determined using a logistic regression classifier for each activity. The results are presented in Table 8.

Table 8: Logistic regression classification for stop-out prediction.

Variable Set	Variables	Accuracy	Kappa Statistics
Score-based	Structured	48.00%	-0.06
	Structured + Unstructured	66.67%	0.43
	MCQ	100.00%	1.00
Sequence-based	Videos	75.00%	0.48
	Structured	81.48%	0.61
	Unstructured	81.82%	0.63
	MCQ	82.35%	0.56

The stop-out prediction accuracy increases as more activity scores are included in the logistic regression models. The accuracy of these models is highly dependent on the inclusion of the MCQ activity scores. The MCQ activity is the last activity students must complete before proceeding to Phase 2.

The decision tree for the video activity, as shown in Figure 4, identify premature termination (ASA V2) as the best criterion for determining if students are likely to stop the activities. Prematurely terminating attempts at a frequency higher than 25% of the student's attempts is predictive of stopping the activity 83% of the time. In addition, a low compliance with incomplete video watching by fast-forwarding (ASA V10) and completing the video without additional actions (ASA V3) are indicative of students who stop out of the learning environment.

For the structured activity, a high compliance with the recommended process but without submitting an answer (ASA S10), answering questions with highlighting of keywords (ASA S5) and answering questions with the scripted steps, as shown in Figure 5, all indicate students who are likely to proceed to Phase 2. Students who tend not to provide an answer for these attempts are likely to not proceed to Phase 2.

While the unstructured activity gives more freedom to participants, the number of splitting criteria is minimal. Learners who do not proceed to Phase 2 are characterized by submitting more than 13% of their questions without any attempt to solve them (ASA U3). Also students who complied more than with the scripted steps more than 56% of the time also tended to stop out (ASA U12). We note that the lower compliance with the RIGHT sequence in unstructured activity in Phase 1 is associated with 86% probability of learners proceeding to Phase 2.

7. Conclusions

In this study, we presented a framework for converting clickstream data into action sequence archetypes. ASAs provide insight into how students approach learning activities by consolidating similar plans of action under a common label. For us, having a common label to refer to different patterns of actions facilitates discussion and interdisciplinary collaboration between the computer sciences and the learning sciences. This collaboration led us away from trying to analyze learning outcomes with click counts and time on task measures and toward ASAs. ASA frequencies identify how often a learner attempts to reach a goal via a particular method.

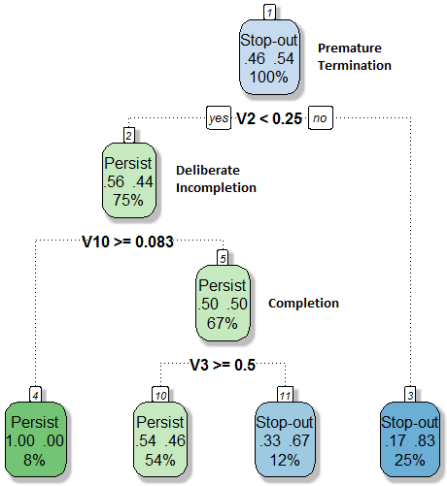


Figure 4: Decision tree for the video activity.

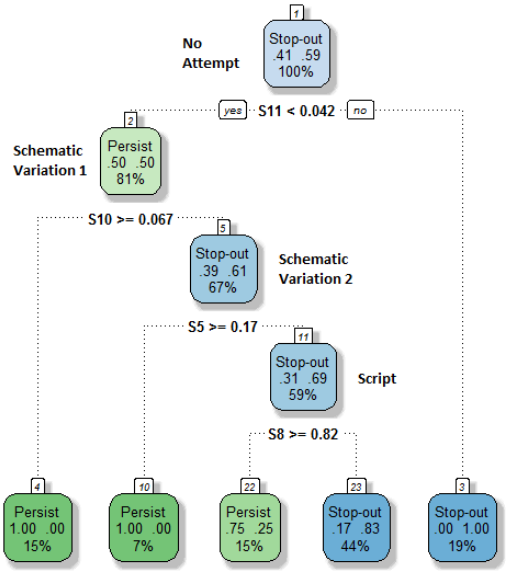


Figure 5: Decision tree for the structured activity.

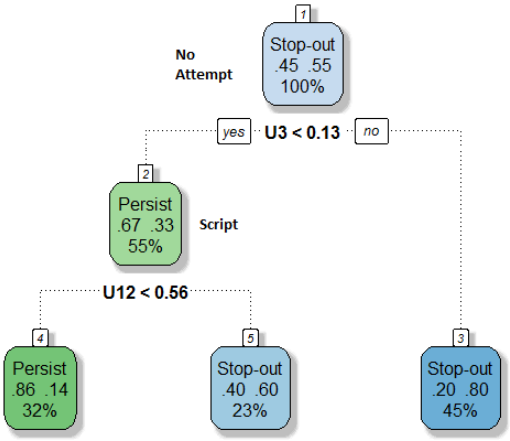


Figure 6: Decision tree for the unstructured activity.

Looking at our decision trees, ASAs can be used to quickly identify whether a learner is using on-task or off-task behaviors. However, they also can also be used to separate different approaches to achieving the same goal.

In our case, students whose action sequences aligned more strongly to the archetype representing the RIGHT sequence presented in the online videos were less likely to persist to the second phase of activities. In one sense, it is counterintuitive to suggest that students who follow a taught script more closely would be less likely to persist in an activity. However, if script use is a way of minimizing cognitive load, novices who consistently exhibit script-like behaviors could be indicating more routinization and less assimilation of new concepts. What these students may have learned from their classroom instruction and the online material is a series of steps for completing the structured and unstructured activities and not the generalizable schema that underlies those activities.

Using the ASAs to separate script users from generalizable schema users gives us a method of predicting a student's likelihood of persisting through the first phase of activities and attempting the second phase composed of unfamiliar math models. This method of prediction identifies students who are likely to stop out before the second phase much earlier than looking at how accurately the students solve the word problems. By the end of the second activity, our model could predict with high accuracy whether a student would continue on to Phase 2. Using a more traditional method of performance assessment and analyzing accuracy levels to predict future behavior required students to complete all of Phase 1 before the model could accurately predict whether the student would persist. In short, using ASAs to analyze how students approach the activities is more diagnostic of future performance than looking at past performance measures.

Finally, it is not lost on us that we developed an algorithm that converts action sequences (scripts) into action sequence archetypes (schemas) to measure students' use of scripts and generalizable schemas. For this project, the machine learning goals and the students learning goals happened to overlap. We plan to continue developing the parallels by integrating our ASA analysis into a student feedback engine that can shift students away from off-task behaviors and toward on-task behaviors. We also seek to lead on-task students toward more productive action sequences that foster the development of generalizable problem solving schemas rather than specific problem solving scripts.

8. ACKNOWLEDGMENTS

This project is supported by a start-up grant from the Centre for Research and Development in Learning (CRADLE@NTU).

9. REFERENCES

[1] Kho, T. H. 1987. Mathematical models for solving arithmetic problems. In Proceedings of the Fourth Southeast Asian Conference on Mathematics Education (ICMI-SEAMS), 345-351.

[2] Polya, G., Kaddouch, R., Renou, M., Comtois, M., and Dubois, M. J. 1957. How to solve it: a new aspect of mathematical method. Princeton University Press, Princeton, NJ.

[3] Abelson, R. P. 1981. Psychological status of the script concept. American Psychologist, 36, 7 (Jul. 1981), 715-729.

[4] Schoenfeld, A. H. 1999. Models of the teaching process. The Journal of Mathematical Behavior, 18, 3, (Mar. 1999), 243-261.

[5] Rumelhart, D. E., and Ortony, A. 1977. The representation of knowledge in memory. In Schooling and the Acquisition of Knowledge, R. C. Anderson, R. J. Spiro, and W. E. Montague, Eds. Erlbaum Associates, Hillsdale, NJ.

[6] Schank, R. C., and Abelson, R. P. 1977. Scripts, Plans, Goals, and Understanding: An inquiry into human knowledge structures. Erlbaum Associates, Hillsdale, NJ.

[7] Sweller, J., and Chandler, P. 1994. Why some material is difficult to learn. Cognition and Instruction, 12, 3, (Sep. 1994), 185-233.

[8] Cheong, Y. K. 2002. The model method in Singapore. The Mathematics Educator, 6, 2, 47-64.

[9] Emond, B., and Scott Buffett, S. 2015. Analyzing student inquiry data using process discovery and sequence classification. In Proceedings of the 8th International Conference on Data Mining, (Atlantic City, NJ, USA, November, 14-17, 2015), ICDM'15. 412-415.

[10] Southavilay, V., Markauskaite, L., and Jacobson, M. J. 2013. From "events" to "activities": Creating abstraction techniques for mining students' model-based inquiry processes. In Proceedings of the 6th International Conference on Educational Data Mining, (Memphis, TN, USA, July 6-9, 2013), EDM'13, 280-283.

[11] Ye, C., Kinnebrew, J. S., Segedy, J. R., and Biswas, G. 2015. Learning behavior characterization with multi-feature, hierarchical activity sequences. In Proceedings of the 8th International Conference on Educational Data Mining, (Madrid, Spain, June, 26-29, 2015), EDM'15, 380-383.

[12] Veeramachaneni, K., O'Reilly, U. M., and Taylor, C. 2014. Towards feature engineering at scale for data from massive open online courses. arXiv preprint arXiv:1407.5238.

[13] Taylor, C., Veeramachaneni, K., and O'Reilly, U. M. 2014. Likely to stop? Predicting stopout in massive open online courses. arXiv preprint arXiv:1408.3382.

[14] Hu, Y. H., Wu, F., and Liao, Y. J. 2013. An efficient tree-based algorithm for mining sequential patterns with multiple minimum supports. Journal of Systems and Software, 86, 5, (May 2013), 1224-1238.

[15] Frey, B. J., and Dueck, D. 2007. Clustering by passing messages between data points. Science, 315, 5814, (Feb. 2007), 972-976.

[16] Islam, A., and Inkpen, D. 2008. Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD), 2, 2, (Jul. 2008), 10.

[17] Sparck Jones, K., 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 28, 1, (Jan. 1972), 11-21.

A Coupled User Clustering Algorithm for Web-based Learning Systems

Ke Niu[*†]
kk511@bit.edu.cn

Zhendong Niu[*✉]
zniu@bit.edu.cn

Xiangyu Zhao[*]
koopr@bit.edu.cn

Can Wang[‡]
can.wang@csiro.au

Kai Kang[*]
kangkai@bit.edu.cn

Min Ye[*]
2120141074@bit.edu.cn

[*]School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
[†]Computer School, Beijing Information Science and Technology University, Beijing, China
[‡]Digital Productivity, Commonwealth Scientific and Industrial Research Organisation, Sandy Bay, Australia

ABSTRACT

User clustering algorithms have been introduced to analyze users' learning behaviors and help to provide personalized learning guides in traditional Web-based learning systems. However, the explicit and implicit coupled interactions, which means the correlations between user attributes generated from learning actions, are not considered in these algorithms. Much significant and useful information which can positively affect clustering accuracy is neglected. To solve the above issue, we proposed a coupled user clustering algorithm for Wed-based learning systems. It respectively takes into account intra-coupled and inter-coupled relationships of learning data, and utilizes Taylor-like expansion to represent their integrated coupling correlations. The experiment result demonstrates the outperformance of the algorithm in terms of efficiently capturing correlations of learning data and improving clustering accuracy.

Keywords

Web-based learning, coupled interactions, user clustering, user behavior analysis

1. INTRODUCTION

Information technology and its application have brought great changes to all aspects of human, especially education area. Web-based learning is a significant and advanced way of education, meaning to utilize computer network technology, digital multimedia technology, database technology and other modern information technology to learn in digital environment. Compared with traditional learning, Web-based learning can efficiently meet learners' needs of learning anytime and anywhere. Meanwhile, it takes advantage of various online resources and helps learners to expand their horizons and discover interests.

Recently Web-based learning systems are studied by many education institutions and researchers, and a large number of online learning communities and virtual schools arise [1]. As an emerging online learning system, MOOC (Massive Open Online Courses) was initiated by America's top universities in 2012. It had a participation of more than 6 million of students from around 220 countries within one year [2]. Some of Web-based learning systems apply user clustering algorithms to analyze learning behaviors and provide personalized learning services. Fu and Ofoghlu put forward a new clustering algorithm; it can extract clusters which can be described by overlapping layered concept in dense space [3]. According to the feedback of basic clustering method, Montazer et al. proposed a hybrid clustering algorithm, which considered clustering issues from different perspectives, and kept the simplicity of basic clustering algorithm [4]. Another matrix-based improved clustering algorithm was put forward by Zhang et al., and it is much more efficient when comparing with K-means [5]. Lin et al. came up with a kind of intuitionistic fuzzy kernel clustering algorithm (KIFCM), combining intuitionistic fuzzy sets and fuzzy kernel clustering algorithm, and applied it in learner behavior analysis [6].

With the above algorithms utilized in Wed-based learning systems, learners' attribute information is extracted by analyzing their behaviors, and finally used for user clustering. However, these algorithms generally neglect the explicit and implicit coupling relationships of user attributes and this may lead to massive significant information loss. For example, table 1 presents an evaluation index system based on information provided by a specified Web-based learning system. With common sense, we think that user attribute of "Average correct rate of homework" has a positive impact on "Comprehensive test result". Generally, if the "Average correct rate of homework" is better, the "Comprehensive test result" is better. Students who behave this way are categorized in "normal" group. However, there are also students who can either get a better "Average correct rate of homework" with a worse "Comprehensive test result", or a better "Comprehensive test result" with a worse "Average correct rate of homework"; they are categorized in "unnormal" group. These unnormal situations are caused by irregular correlations of user attributes, but they are often ignored. This will certainly have negative effect on user clustering

Table 1: Comprehensive evaluation index system

First-level index	Second-level index
Autonomic learning	Times of doing homework
	Average correct rate of homework
	Number of learning resources
	Total time length of learning resources
	Times of daily quiz
	Daily average quiz result
	Comprehensive test result
	Number of collected resources
	Times of downloaded resources
	Times of making notes
Interactive learning	Times of asking questions
	Times of marking and remarking
	Times of answering classmates' questions
	Times of posting comments on the BBS
	Times of interaction by BBS message
	Times of sharing resources
	Average marks made by the teacher
	Average marks made by other students
	Times of marking and remarking made by the student for the teacher
	Times of marking and remarking made by the student for other students

accuracy.

Nowadays an increasing number of researchers are studying the interactions between object attributes with special attention and have been aware that the independence assumption on attributes often leads to a mass of information loss. In addition to the basic Pearson's correlation [7], Wang et al. put forward the intra-coupled and inter-coupled interactions of continuous attributes [8]. An innovative coupled group-based matrix factorization model for discrete attributes of recommender system was addressed by Li et al. [9]. Jakulin and Bratko proposed an algorithm to detect interactions between attributes, but it is only applicable in supervised learning with the experimental results [10]. For unsupervised learning, the coupled nominal similarity to extract new relationships between entities was addressed by Wang et al., but it is only for categorical data [11]. We rarely find any methods applied in Web-based learning systems, that consider coupling relationships of user attributes in user clustering.

This paper proposed a coupled user clustering algorithm for Web-based learning systems, namely CUCA. It studies the coupling relationships of user attributes. With the help of Taylor-like expansion, we use a spectral clustering algorithm to cluster users. When it is applied in Web-based learning systems, it can efficiently capture learners' behavior features and analyze the information behind them, especially that of "unnormal" group of learners, and finally use them to provide personalized learning services. To verify the outperformance of CUCA, we compare its clustering result with that of 3 other algorithms, respectively from 3 dimensions of learning attitude, learning effect and the integrated dimension.

The rest of the paper is organized as following. The clustering algorithm model is proposed in section 2. Section 3 introduces the formalization and exemplification of the clustering algorithm. In section 4, experiments and results analysis are demonstrated. Section 5 concludes the paper and discusses some potential applications of the proposed algorithm in the future.

2. CLUSTERING MODEL

Evaluation model usually plays the core role in user evaluation framework [12]. In this section, the coupled user clustering model is illustrated. This model captures coupling relationships of user attributes through online behavior analysis, and uses spectral clustering algorithm to improve clustering accuracy.

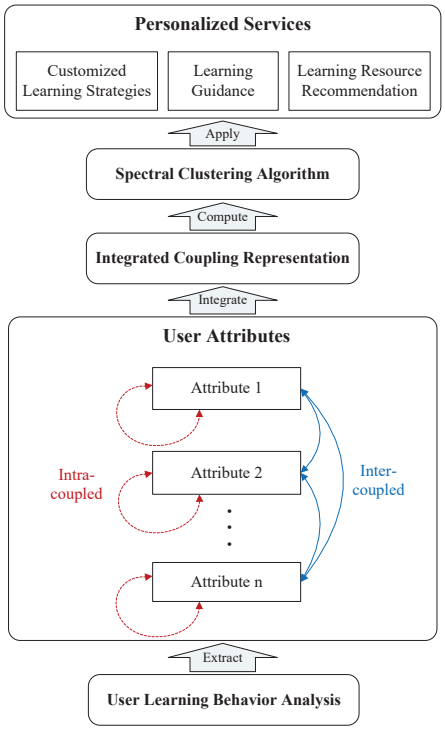


Figure 1: The coupled user clustering model

The model is composed of user learning behavior analysis, coupled interactions computation of user attributes, integrated coupling representation and spectral clustering algorithm, illustrated in figure 1. As the basis, data for user learning behavior analysis needs to be collected, consolidated and normalized. From the data, user attributes information is extracted. With the extracted user attributes, the intra-coupled interaction within an attribute and inter-coupled interaction among different attributes are respectively captured. Then all the interactions are integrated and represented using Taylor-like expansion. Finally we use a spectral clustering algorithm - NJW to cluster users. This model is consequently applied in various Web-based personalized services, like Learning guide customization, tutoring and learning resources recommendation.

3. CLUSTERING ALGORITHM

Based on the model illustrated in section 2, this paper proposed an online coupling user clustering algorithm. It is

Table 2: A fragment example of user attributes

U \ A	a_1	a_2	a_3	a_4	a_5	a_6
u_1	0.61	0.55	0.47	0.72	0.63	0.62
u_2	0.75	0.92	0.62	0.63	0.74	0.74
u_3	0.88	0.66	0.71	0.74	0.85	0.87
u_4	0.24	0.83	0.44	0.29	0.21	0.22
u_5	0.93	0.70	0.66	0.81	0.95	0.93

suitable for network education, not only applicable to user clustering analysis in Web-based learning systems, but also to enterprise training, performance review and others with users participation and behaviors recording. This section describes the details of the proposed coupled user clustering algorithm. Firstly, it collects user learning behavior information and extracts user attributes from them. Secondly, it calculates and represents users' intra-coupled and inter-coupled relationship. Thirdly, the intra-coupled and inter-coupled interactions are integrated to be a coupled representation. Finally, it clusters users based on the processed attributes, using NJW algorithm.

3.1 User learning behavior analysis

When students login a Web-based learning system, the system will record their activity information, such as number of learning resources, total time length of learning resources and average correct rate of homework, which can be used to build an evaluation index system. We refer to a Web-based personalized user evaluation model [13] and utilizes its evaluation index system to extract students' attributes information. This index system is with evaluation standards of America K-12 (kindergarten through twelfth grade) [14] and Delphi method [15], which is a hierarchical structure built according to mass of information and data generated during general e-learning activities. It defines 20 indicators and can comprehensively represent the students' attributes, as shown in table 1.

Generally attributes are with various data types and units, we formalize them by creating the table 2.

3.2 Intra-coupled and inter-coupled representation

In this section, we represent intra-coupled and inter-coupled interactions of user attributes. And with a few examples, the application of CUCA is demonstrated. We choose 5 students and 6 of the 20 attributes in table 1, which are "Average correct rate of homework", "Times of doing homework", "Number of learning resources", "Total time length of learning resources", "Daily average quiz result" and "Comprehensive test result". The 6 attributes are respectively signified by a_1, a_2, a_3, a_4, a_5 and a_6 in table 2.

Here we use a tetrad $S = \langle U, A, V, f\rangle$ to represent user attributes information. $U = \{u_1, u_2, \ldots, u_m\}$ means a finite set of users; $A = \{a_1, a_2, \ldots, a_n\}$ refers to a finite set of attributes; $V = \bigcup_{j=1}^{n} V_j$ represents all attributes value sets; $V_j = \{a_j{\cdot}v_1, \ldots, a_j{\cdot}v_{t_j}\}$ is the value set of the j-th attribute; $f = \bigcup_{i=1}^{n} f_j$, $f_j : U \to V_j$ is the function for calculating a certain attribute value. For example, the information table 2 above contains 5 users $\{u_1, u_2, u_3, u_4, u_5\}$ and 6 attributes $\{a_1, a_2, a_3, a_4, a_5, a_6\}$; the first attribute value of u_1 is $f_1(u_1) = 0.61$.

The common way to calculate the interactions between 2 attributes is Pearson's correlation coefficient [7]. The user attributes from the Table 1 are continuous variables and approximate to Normal distribution, meeting the constraint condition of the Pearson's correlation coefficient. Thus we use it to help to calculate attributes interactions in this paper. For instance, the Pearson's correlation coefficient between a_k and a_j is formalized as:

$$Cor(a_j, a_k) = \frac{\sum_{u\in U}(f_j(u) - \mu_j)(f_k(u) - \mu_k)}{\sqrt{\sum_{u\in U}(f_j(u) - \mu_j)^2 \sum_{u\in U}(f_k(u) - \mu_k)^2}} \tag{1}$$

Where μ_j, μ_k are respectively mean values of a_j, a_k.

The Pearson's correlation coefficient helps to calculate the attributes interactions, but it fits for linear relationship only, which is not sufficient to fully capture pairwise attributes interactions. Therefore we converts the original data attributes into a higher dimensional feature space to extract more attribute information [16].

Firstly, we use a few additional attributes to expand interaction space. Then there are L attributes for each original attribute a_j, including itself, namely $\langle a_j\rangle^1$, $\langle a_j\rangle^2$, ... , $\langle a_j\rangle^L$. Each attribute value is the power of the attribute, for instance, $\langle a_j\rangle^3$ is the third power of attribute a_j, $\langle a_j\rangle^p$ $(1 \le p \le L)$ is the p-th power of a_j. In table 3, the denotation a_j and $\langle a_j\rangle^1$ are equivalent; the value of $\langle a_j\rangle^2$ is the square of that of a_j. For simplicity, we set L=2.

Secondly, the correlation between pairwise attributes is calculated. It captures both local and global coupling relations. We take the p-values for testing the hypotheses of no correlation between attributes into account. p-value here means the probability of getting the maximum correlation observed by random chance, while the true correlation is zero. If p-value is smaller than 0.05, the correlation $Cor(a_j, a_k)$ is significant. The updated correlation coefficient is as:

$$R\_Cor(a_j, a_k) = \begin{cases} Cor(a_j, a_k) & \text{if } p\text{-value}<0.05, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Here we do not consider all relationships, but only takes the significant coupling relationships into account, because all relationships involvement may cause the over-fitting issue on modeling coupling relationship. This issue will go against the attribute inherent interaction mechanism. So based on the updated correlation, the intra-coupled and inter-coupled interaction of attributes is proposed. Intra-coupled interaction is the relationship between a_j and all its powers; inter-coupled interaction is the relationship between a_j and powers of the rest attributes a_k $(k \ne j)$.

Table 3: Extended user attributes

U \ $\widetilde{A}$	$\langle a_1\rangle^1$	$\langle a_1\rangle^2$	$\langle a_2\rangle^1$	$\langle a_2\rangle^2$	$\langle a_3\rangle^1$	$\langle a_3\rangle^2$	$\langle a_4\rangle^1$	$\langle a_4\rangle^2$	$\langle a_5\rangle^1$	$\langle a_5\rangle^2$	$\langle a_6\rangle^1$	$\langle a_6\rangle^2$
u_1	0.61	0.37	0.55	0.30	0.47	0.22	0.72	0.52	0.63	0.40	0.62	0.38
u_2	0.75	0.56	0.92	0.85	0.62	0.38	0.63	0.40	0.74	0.55	0.74	0.55
u_3	0.88	0.77	0.66	0.44	0.71	0.50	0.74	0.56	0.85	0.72	0.87	0.76
u_4	0.24	0.06	0.83	0.69	0.44	0.19	0.29	0.08	0.21	0.04	0.22	0.05
u_5	0.93	0.86	0.70	0.49	0.66	0.44	0.81	0.66	0.95	0.90	0.93	0.86

Definition 1 Intra-coupled interaction. The intra-coupled interaction within an attribute is represented as a matrix. For attribute a_j, it is an $L \times L$ matrix $R^{Ia}(a_j)$. In the matrix, (p, q) is the correlation between $\langle a_j\rangle^p$ and $\langle a_j\rangle^q$ $(1 \le p, q \le L)$.

$$R^{Ia}(a_j) = \begin{pmatrix} \alpha_{11}(j) & \alpha_{12}(j) & \cdots & \alpha_{1L}(j) \\ \alpha_{21}(j) & \alpha_{22}(j) & \cdots & \alpha_{2L}(j) \\ \cdots & \cdots & \ddots & \cdots \\ \alpha_{L1}(j) & \alpha_{L2}(j) & \cdots & \alpha_{LL}(j) \end{pmatrix} \quad (3)$$

Where $\alpha_{pq}(j) = R\_Cor(\langle a_j\rangle^p, \langle a_j\rangle^q)$ is the Pearson's correlation coefficient between $\langle a_j\rangle^p$ and $\langle a_j\rangle^q$.

For attribute a_1 in table 3 above, we can get the intra-coupled interaction of it as $R^{Ia}(a_1) = \begin{pmatrix} 1 & 0.986 \\ 0.986 & 1 \end{pmatrix}$, which means that the correlation coefficient between attribute "Average correct rate of homework" and its second power is as high as 0.986. There is close relationship between them.

Definition 2 Inter-coupled interaction. The inter-coupled interaction between attribute a_j and other attributes a_k $(k \ne j)$ is quantified as an $L \times L * (n-1)$ matrix as:

$$R^{Ie}(a_j|\{a_k\}_{k\ne j}) = \begin{pmatrix} R^{Ie}(a_j|a_{k_1}) & \cdots & R^{Ie}(a_j|a_{k_{n-1}}) \end{pmatrix} \quad (4)$$

$$R^{Ie}(a_j|a_{k_i}) = \begin{pmatrix} \beta_{11}(j|k_i) & \beta_{12}(j|k_i) & \cdots & \beta_{1L}(j|k_i) \\ \cdots & \cdots & \ddots & \cdots \\ \beta_{L1}(j|k_i) & \beta_{L2}(j|k_i) & \cdots & \beta_{LL}(j|k_i) \end{pmatrix} \quad (5)$$

Here $\{a_k\}_{k\ne j}$ refers to all the attributes except for a_j, and $\beta_{pq}(j|k_i) = R\_Cor(\langle a_j\rangle^p, \langle a_{k_i}\rangle^q)$ is the correlation coefficient between $\langle a_j\rangle^p$ and $\langle a_{k_i}\rangle^q$ $(1 \le p, q \le L)$.

For attribute a_1 in the table 3 above, the inter-coupled interaction between a_1 and others $(a_2, a_3, a_4, a_5, a_6)$ is calculated as:

$$R^{Ie}(a_1|\{a_2, a_3, a_4, a_5, a_6\}) =$$

$$\begin{pmatrix} 0 & 0 & 0.898 & 0.885 & 0.928 & 0.921 \\ 0 & 0 & 0.929 & 0.920 & 0.879 & 0.888 \end{pmatrix}$$

$$\begin{pmatrix} 0.997 & 0.982 & 0.999 & 0.988 \\ 0.978 & 0.994 & 0.982 & 0.999 \end{pmatrix}$$

The p-values between a_1 and others $(a_2, a_3, a_4, a_5, a_6)$ is calculated as:

$$p^{Ie}(a_1|\{a_2, a_3, a_4, a_5, a_6\}) =$$

$$\begin{pmatrix} 0.689 & 0.677 & 0.039 & 0.046 & 0.023 & 0.027 \\ 0.733 & 0.707 & 0.023 & 0.027 & 0.050 & 0.044 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0.003 & 0 & 0.002 \\ 0.004 & 0.001 & 0.003 & 0 \end{pmatrix}$$

Based on the result, we can find that there is hidden correlation between user attributes. For instance, all the p-values between attribute a_1 and a_2 are larger than 0.05, so the correlation coefficient is 0 based on Equation (2), indicating there is no significant correlation between "Average correct rate of homework" and "Times of doing homework". Meanwhile, the correlation coefficient between a_1 and a_5, a_1 and a_6 is quite close to 1; it indicates "Daily average quiz result" and "Comprehensive test result" respectively have close relationship with "Average correct rate of homework", which is consistent with our practical experiences. In conclusion, comprehensively taking into account intra-coupled and inter-coupled correlation of attributes can efficiently help capturing coupling relationships between user attributes.

3.3 Integrated coupling representation

Intra-coupled and inter-coupled interactions are integrated in this section as a coupled representation scheme.

In table 3 above, each user is signified by $L*n$ updated variables $\widetilde{A} = \{\langle a_1\rangle^1, \ldots, \langle a_1\rangle^L, \ldots, \langle a_n\rangle^1, \ldots, \langle a_n\rangle^L\}$. With the updated function $\widetilde{f}_j^p(u)$, the corresponding value of attribute $\langle a_n\rangle^p$ is assigned to user u. Attribute a_j and all its powers are signified as $\widetilde{u}(a_j) = [\widetilde{f}_j^1(u), \ldots, \widetilde{f}_j^L(u)]$, while the rest attributes and all powers are presented in another vector $\widetilde{u}(\{a_k\}_{k\ne j}) = [\widetilde{f}_{k_1}^1(u), \ldots, \widetilde{f}_{k_1}^L(u), \ldots, \widetilde{f}_{k_{n-1}}^1(u), \ldots, \widetilde{f}_{k_{n-1}}^L(u)]$. For instance, in table 3, $\widetilde{u_1}(a_1) = [0.61, 0.37]$, $\widetilde{u_1}(\{a_2, a_3, a_4, a_5, a_6\}) = [0.55, 0.30, 0.47, 0.22, 0.72, 0.52, 0.63, 0.40, 0.62, 0.38]$.

Definition 3 Coupled representation. Attribute a_j's coupled representation is formalized as a $1\times L$ vector $u^c(a_j|\widetilde{A}, L)$,

where $(1, p)$ component corresponds to the updated attribute $\langle a_j \rangle^p$.

$$u^c(a_j|\widetilde{A}, L) = u^{Ia}(a_j|\widetilde{A}, L) + u^{Ie}(a_j|\widetilde{A}, L) \tag{6}$$

$$u^{Ia}(a_j|\widetilde{A}, L) = \widetilde{u}(a_j) \odot w \otimes [R^{Ia}(a_j)]^T \tag{7}$$

$$\begin{aligned} u^{Ie}(a_j|\widetilde{A}, L) = \widetilde{u}(\{a_k\}_{k\neq j}) \odot [w, w, \ldots, w] \\ \otimes [R^{Ie}(a_j|\{a_k\}_{k\neq j})]^T \end{aligned} \tag{8}$$

where $w = [1/(1!), 1/(2!), \ldots, 1/(L!)]$ is a constant $1 \times L$ vector, $[w, w, \ldots, w]$ is a $1 \times L*(n-1)$ vector concatenated by $n-1$ constant vectors w. $\odot$ denotes the Hadamard product, and $\otimes$ represents the matrix multiplication.

Take an example in table 4, the coupled representation for attribute a_1 is presented as $u_1^c(a_1|\widetilde{A}, 2) = [3.85, 3.80]$. The reason we choose such a representation method is explained below. If the above Equation (6) is expanded, for example, we get the $(1, p)$ element which corresponds to $\langle a_j \rangle^p$ of the vector $u^c(a_j|\widetilde{A}, L)$ as below, which resembles Taylor-like expansion of functions [17].

$$\begin{aligned} u^c(a_j|\widetilde{A}, L) \cdot \langle a_j \rangle^p = \alpha_{p1}(j) \cdot \widetilde{f}_j^1(u) + \sum_{i=1}^{n-1} \frac{\beta_{p1}(j|k_i)}{1!} \widetilde{f}_{k_i}^1(u) \\ + \frac{\alpha_{p2}(j)}{2!} \widetilde{f}_j^2(u) + \sum_{i=1}^{n-1} \frac{\beta_{p2}(j|k_i)}{2!} \widetilde{f}_{k_i}^2(u) + \ldots \\ + \frac{\alpha_{pL}(j)}{L!} \widetilde{f}_j^L(u) + \sum_{i=1}^{n-1} \frac{\beta_{pL}(j|k_i)}{L!} \widetilde{f}_{k_i}^L(u) \end{aligned} \tag{9}$$

Finally we obtained the global coupled representation of all the n original attributes as a concatenated vector:

$$u^c(\widetilde{A}, L) = [u^c(a_1|\widetilde{A}, L), u^c(a_2|\widetilde{A}, L), \ldots, u^c(a_n|\widetilde{A}, L)] \tag{10}$$

With the couplings of attributes, each user is represented as a $1 \times L*n$ vector. When all the users follow the steps above, we then obtain an $m \times L*n$ coupled information table. For example, based on table 2, the coupled information table shown in table 4, is the new representation.

3.4 User clustering

We obtained the global coupled representation in table 4. Compared with the original representation, this one reflects coupling interactions of attributes, and contains far more coupling relationships. With these data, we can do user clustering using NJW [18], which is a kind of spectral clustering algorithm. Detailed clustering results are demonstrated in experiment later.

4. EXPERIMENTS AND EVALUATION

In this section, we conduct experiments to verify the validity and accuracy of the proposed algorithm. The data for the experiments are collected from a Web-based learning system of China Educational Television (CETV), named "New Media Learning Resource Platform for National Education"[1]. As a basic platform for national lifelong education, which started the earliest in China, and had the largest group of users and provided most extensive learning resources, it met the needs of personalization and user diversity through integrating a variety of multi-network, terminals and resources. So far, the number of registered users has reached more than two million. The experiment is composed of 3 parts: user study, user clustering and result analysis.

4.1 User study

In the experiment, we ask 220 users (signified by $s_1, s_2, \ldots, s_{220}$) to learn C programming language online. The whole learning process, including recording and analyzing learning activities information, is accomplished in CETV.

The public data sets regarding learners' learning behaviors in online learning systems are insufficient, and most of them don't contain labeled user clustering information. Meanwhile, because learners always behave with certain subjectivity in online learning systems, to label learners with different classifiers based on their learning behaviors only, but without the information behind, is not accurate. Therefore, we adopt a few user study methods, including self-assessment, peer-assessment and teacher-assessment [19], to label online learners with classifiers. It is the basis for verifying the accuracy of clustering.

Analyzing the 20 attributes extracted from table 1 using user evaluation index system proposed in this paper, we can easily find that they can be mainly divided into 2 categories. Some attributes belong to the category of "learning attitude", which refers to students' learning initiatives, like "Times of doing homework", "Number of learning resources" and "Total time length of learning resources". While the rest belong to the category of "learning effect", which refers to how well students receive knowledge, like "Average correct rate of homework", "Daily average quiz result" and "Comprehensive test result". Accordingly, we can label learners with these attributes from both categories. Each of the attributes has 3 grades - high, medium and low. Consequently every learner has 2 labels and each label has one grade of high, medium and low. In total, there will be 9 different combinations - high & high, high & medium, high & low, medium & high, medium & medium, medium & low, low & high, low & medium and low & low.

After the students had finished a learning phase, we asked the 220 users to do a self-assessment using centesimal grade, respectively from perspectives of learning attitude and learning effect. Then we requested teacher assessments in the same way, meaning the teacher of the subject to review the students' performance. Finally, the students were asked to do peer-assessments, which means students do an assessment for each other. Each student will get the assessment scores from the rest 219 students. We calculate the aver-

[1]http://www.guoshi.com/

Table 4: Integrated coupling representation of user attributes

U \ $\widetilde{A}$	$\langle a_1\rangle^1$	$\langle a_1\rangle^2$	$\langle a_2\rangle^1$	$\langle a_2\rangle^2$	$\langle a_3\rangle^1$	$\langle a_3\rangle^2$	$\langle a_4\rangle^1$	$\langle a_4\rangle^2$	$\langle a_5\rangle^1$	$\langle a_5\rangle^2$	$\langle a_6\rangle^1$	$\langle a_6\rangle^2$
u_1	3.85	3.80	0.70	0.70	2.20	1.46	3.24	3.23	3.35	3.70	3.76	3.81
u_2	4.54	4.50	1.34	1.34	2.89	1.98	3.66	3.65	3.82	4.31	4.37	4.51
u_3	5.51	5.46	0.88	0.88	3.54	2.44	4.46	4.45	4.66	5.22	5.28	5.47
u_4	1.53	1.52	1.17	1.17	1.01	0.80	1.03	1.02	1.06	1.42	1.44	1.52
u_5	5.94	5.89	0.94	0.94	3.73	2.49	4.95	4.94	5.17	5.68	5.75	5.90

Table 5: Transformation rule between score and grade

Score range	Grade	Sample
$80 \leq X \leq 100$	high	95
$50 \leq X < 80$	medium	75
$0 \leq X < 50$	low	40

Table 6: The evaluation results of s_1

	learning attitude	learning effect
Self-assessment (40%)	80.0	75.0
Teacher-assessment (35%)	85.0	80.0
Peer-assessment (25%)	82.7	79.2
Comprehensive evaluation results	82.4	77.8
grade	high	medium
Class	high & medium	

age of the 219 scores. A student's final score is obtained by integrating the 3 assessments above. According to Expert Investigation Weight Method [15], we did statistical analysis and got approximate weights for the assessments, namely 40% for self-assessment, 35% for teacher-assessment and 25% for peer-assessment. Each student's final score will be transformed into a grade value, "high", "medium" or "low". The transformation rule between score and grade is shown in table 5.

Take student s_1 as an example, his 3 assessment scores and transformed grades are shown in table 6.

4.2 User clustering

In Equation (9), the proposed coupled representation is strongly dependent on how large L can be. Thus we conduct a few experiments to study how the performance of L influences the clustering accuracy of CUCA. The range of L value is from $L = 1$ to $L = 10$. With the growth of L value, $L!$ value grows. When L = 10, it is large enough to capture most of the information in Equation (9). The experiments show that with the growth of L, the clustering accuracy will be gradually improved. When $L = 3$, the accuracy change reaches a comparatively stable status; when $L > 3$, the accuracy change is extremely small. That means the accuracy of when $L = 3$ and when $L = 10$ is quite similar. To guarantee the accuracy of experimental results and reduce the complexity of the algorithm, we take $L = 3$ in the following comparative experiments.

In the experiments, we utilize the attributes data generated from the 220 students' learning process, as the basis for clustering. Then we persistently collect data from the process which reaches 30 hours by average. Respectively with the help of K-means algorithm, Fuzzy C-means algorithm(FCM), NJW algorithm and CUCA algorithm, we do user clustering, getting 2 labels in terms of learning attitude and learning effect for each student. In section 4.1, we classified each student with 2 labels based on user study result. Then we compare the labels got from user study and user clustering result. If only one label from each side is the same, the clustering accuracy rate is 50%; if both the labels are the same, the accuracy rate reaches 100%. For instance, student s_1 is labeled with "high & medium" in user study, if he is classified to "medium & medium" cluster, the clustering accuracy rate is 50%; if he is classified to "high & medium" cluster, the accuracy rate reaches 100%.

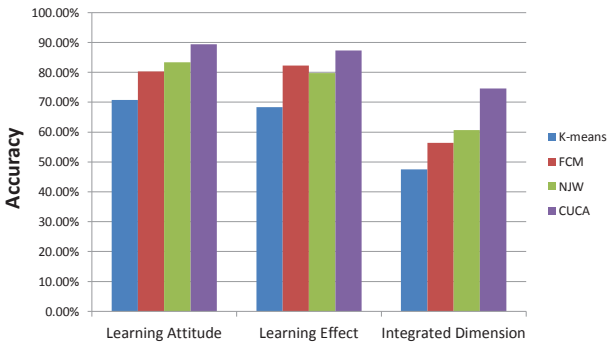
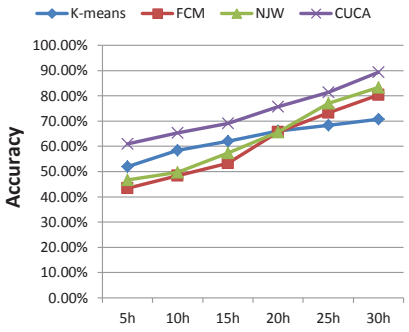


Figure 2: Clustering result analysis (30h)

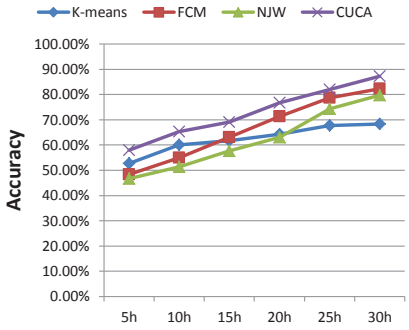
4.3 Result analysis

We do comparison analysis on the clustering result respectively from the 3 dimensions of learning attitude, learning effect and the integrated dimension. The analysis result is shown in figure 2. We can see the clustering accuracy of utilizing CUCA is 89.4% for learning attitude, 87.3% for learning effect and 74.6% for integrated dimension, each of which is higher than that with the other 3 algorithms. Especially, CUCA obviously outperforms the rest on clustering accuracy of integrated dimensions. Compared with K-means, which performs the worst, CUCA improves almost 30% on

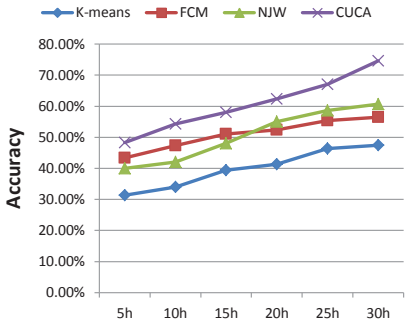
the clustering accuracy. The reason is CUCA fully takes into account coupling relationships of users. In Web-based learning systems, if the user attributes are more complicated, there will be more clustering dimensions and the clustering accuracy will be improved more.



(a) Learning Attitude



(b) Learning Effect

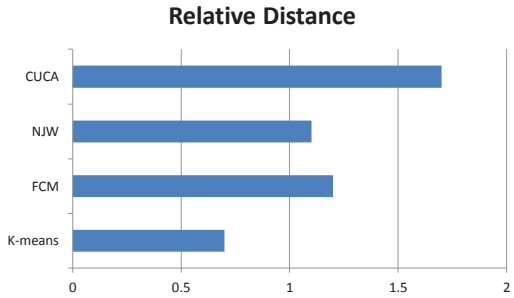


(c) Integrated Dimension

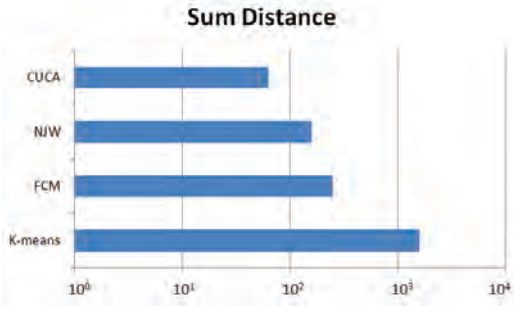
Figure 3: Clustering result of different time phases

If we divide the process of extracting user attributes to 6 phases, namely 5h, 10h, 15h, 20h, 25h, 30h based on average learning length, we can get the correlation between average learning length and clustering accuracy, as shown in figure 3. From the figure, we can see that while the learning length grows, the clustering accuracy of the 4 algorithms keeps improving, specifically for CUCA. With CUCA, the clustering accuracy on integrated dimensions distinctly outperforms that of the 3 other algorithms. It indicates that with the increasing learning behavior data volume, CUCA can find the hidden coupling relationships of user attributes more easily, and the clustering accuracy is much better.

Besides, we can verify clustering accuracy through analyzing user clustering results. The best performance of a clustering algorithm is keeping the distance within clusters as small as possible and the distance between clusters as large as possible. We use the evaluation criteria of Relative Distance (the ratio of average inter-cluster distance upon average intra-cluster distance) and Sum Distance (the sum of object distances within all the clusters) to present the distance. The larger Relative Distance is and the smaller Sum Distance is, the better clustering results are. From figure 4, we can see that the Relative Distance for CUCA is larger than that of the 3 other algorithms, while the Sum Distance for CUCA is smaller. It indicates that CUCA outperforms the rest in terms of clustering structure.



(a) Relative Distance



(b) Sum Distance

Figure 4: Clustering structure analysis (30h)

5. CONCLUSION

A coupled user clustering algorithm (CUCA) for Web-based learning systems is proposed in this paper to capture coupling relationships of user attributes. The algorithm respectively takes intra-coupled and inter-coupled correlation into account in the application process, and utilizes Taylor-like expansion to represent the coupling relationship. Finally, with the usage of spectral clustering algorithm, CUCA is applied to do user clustering. In the experiments, user study, user clustering and result analysis are adopted to verify that CUCA outperforms traditional algorithm for user clustering.

In this paper, the user attributes extracted from user learning behavior data are all numerical data, most of which are continuous data. In reality, there are also categorical data, which will be a significant study topic in the future.

6. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (Project No. 61370137), the National "973" Project of China (No. 2012CB720702) and Major Science and Technology Project of Press and Publication (No: GAPP_ZDKJ_BQ/01).

7. REFERENCES

[1] S. Cai and W. Zhu, "The impact of an online learning community project on university chinese as a foreign language students' motivation," *Foreign Language Annals*, vol. 45, no. 3, pp. 307–329, 2012.

[2] M. Ghosh, "Mooc m4d: An overview and learner's viewpoint on autumn 2013 course." *iJIM*, vol. 8, no. 1, pp. 46–50, 2014.

[3] H. Fu and M. Ó. FoghlÙ, "A conceptual subspace clustering algorithm in e-learning," in *Advanced Communication Technology, 2008. ICACT 2008. 10th International Conference on*, vol. 3. IEEE, 2008, pp. 1983–1988.

[4] G. A. Montazer and M. S. Rezaei, "A new approach in e-learners grouping using hybrid clustering method," in *Education and e-Learning Innovations (ICEELI), 2012 International Conference on*. IEEE, 2012, pp. 1–5.

[5] K. Zhang, L. Cui, H. Wang, and Q. Sui, "An improvement of matrix-based clustering method for grouping learners in e-learning," in *Computer Supported Cooperative Work in Design, 2007. CSCWD 2007. 11th International Conference on*. IEEE, 2007, pp. 1010–1015.

[6] K. Lin, C. Lin, K. Hung, Y. Lu, and P. Pai, "Developing kernel intuitionistic fuzzy c-means clustering for e-learning customer analysis," in *Industrial Engineering and Engineering Management (IEEM), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1603–1607.

[7] G. Gan, C. Ma, and J. Wu, *Data clustering: theory, algorithms, and applications*. Siam, 2007, vol. 20.

[8] C. Wang, Z. She, and L. Cao, "Coupled attribute analysis on numerical data." in *IJCAI*, 2013.

[9] F. Li, G. Xu, L. Cao, X. Fan, and Z. Niu, "Cgmf: coupled group-based matrix factorization for recommender system," in *Web Information Systems Engineering–WISE 2013*. Springer, 2013, pp. 189–198.

[10] A. Jakulin and I. Bratko, *Analyzing attribute dependencies*. Springer, 2003.

[11] C. Wang, L. Cao, M. Wang, J. Li, W. Wei, and Y. Ou, "Coupled nominal similarity in unsupervised learning," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 973–978.

[12] K. Niu, W. Chen, Z. Niu, P. Gu, Y. Li, and Z. Huang, "A user evaluation framework for web-based learning systems," in *Proceedings of the third international ACM workshop on Multimedia technologies for distance learning*. ACM, 2011, pp. 25–30.

[13] K. Niu, Z. Niu, D. Liu, X. Zhao, and P. Gu, "A personalized user evaluation model for web-based learning systems," in *Tools with Artificial Intelligence (ICTAI), 2014 IEEE 26th International Conference on*. IEEE, 2014, pp. 210–216.

[14] L. Katehi, G. Pearson, and M. Feder, *Engineering in K-12 Education: Understanding the Status and Improving the Prospects*. National Academies Press, 2009.

[15] C. Okoli and S. D. Pawlowski, "The delphi method as a research tool: an example, design considerations and applications," *Information & management*, vol. 42, no. 1, pp. 15–29, 2004.

[16] D. Li and C. Liu, "Extending attribute information for small data set classification," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 24, no. 3, pp. 452–464, 2012.

[17] Y. Jia and C. Zhang, "Instance-level semisupervised multiple instance learning." in *AAAI*, 2008, pp. 640–645.

[18] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[19] C. Chang, K. Tseng, and S. Lou, "A comparative analysis of the consistency and difference among teacher-assessment, student self-assessment and peer-assessment in a web-based portfolio assessment environment for high school students," *Computers & Education*, vol. 58, no. 1, pp. 303–320, 2012.

Execution Traces as a Powerful Data Representation for Intelligent Tutoring Systems for Programming

Benjamin Paaßen
CITEC center of excellence
Inspiration 1
33619 Bielefeld, Germany
bpaassen@techfak.uni-bielefeld.de

Joris Jensen
CITEC center of excellence
Inspiration 1
33619 Bielefeld, Germany
jjensen@techfak.uni-bielefeld.de

Barbara Hammer
CITEC center of excellence
Inspiration 1
33619 Bielefeld, Germany
bhammer@techfak.uni-bielefeld.de

ABSTRACT

The first intelligent tutoring systems for computer programming have been proposed more than 30 years ago, mostly focusing on well defined programming tasks e.g. in the context of logic programming. Recent systems also teach complex programs, where explicit modelling of every possible program and mistake is no longer possible. Such systems are based on data-driven approaches, which focus on the syntax of a program or consider the output for example cases. However, the system's understanding of student programs could be enriched by a deeper focus on the actual execution of a program. This requires a suitable data representation which encodes information of programming style as well as its functionality in a suitable way, thus offering entry points for automated feedback generation.

In this contribution we propose a representation of computer programs via execution traces for example input and demonstrate the power of this representation in three key challenges for intelligent tutoring systems: identifying the underlying solution strategy, identifying erroneous solutions and locating the errors in erroneous programs for feedback display.

Keywords

execution traces, data-driven tutoring systems, computer science teaching, sequence alignment, sorting programs

1. INTRODUCTION

Teaching computer programming has been a long-standing goal of intelligent tutoring systems research. The earliest example, the LISP tutor, has been released in 1985 [1] and since then many different approaches have evolved, such as learning by examining and manipulating examples, by simulation and debugging, by dialogue with the system, by collaboration with peers or by feedback [7]. Most of these approaches rely on extensive domain knowledge about program structure, typical mistakes (so-called *buggy rules*) and syntactic concepts, which is expensive to obtain and difficult to encode [5, 10]. In particular, such approaches get infeasible if the space of possible programs (and mistakes) gets too large, and if the goal of the computer program is ill-defined [8]. To push the boundaries of intelligent tutoring systems towards such scenarios, data-driven approaches have been developed which provide feedback to students based on example programs handed in by other students, e.g. by highlighting the difference of the student solution and a similar, correct program [2, 16]. However, such approaches focus strongly on the *syntax* of programs, which is problematic because the relation between a programs functionality and its syntax is highly non-linear.

As an example, consider the Java code shown in Figure 1. The programs on the left and on the middle are both (correct) sorting programs, which have a very similar syntactic structure. Both sort the array via two nested loops, compare the current element to its successor and swap them if the order is incorrect. However, the programs implement *different* algorithms, namely *BubbleSort* (left) and *InsertionSort* (middle). Thus, minor syntactic changes correspond to major changes in terms of function [14]. If an intelligent tutoring system provides feedback based on a functionally dissimilar example (e.g. a different underlying algorithm) the system might recommend changes to the student's program which lead the learner away from her intended strategy. Such feedback might be detrimental to the student's learning success.

This poses a challenge to educational datamining research. How do we estimate the similarity between programs on a functional level, without exceeding effort in knowledge engineering? We propose to represent computer programs by their execution traces, to compare such traces using sequence alignment and to define the similarity between programs based on the alignment distance. An execution trace is a sequence of variable states for each step of the program's execution for some input. They are a usual representation of computer programs for debugging purposes and can provide insight into the dynamic behaviour of programs [6]. In particular, traces and alignments of traces have been successfully applied in educational programming environments to offer students an alternative view on their own program for self-reflection [17, 18]. We build upon this research by utilizing the trace representation for educational datamining,

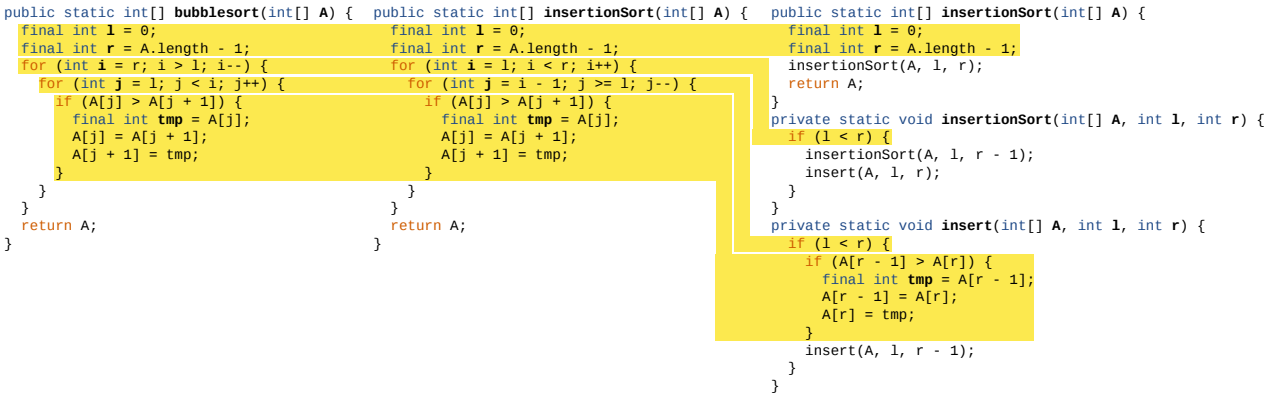


Figure 1: Three correct sorting programs in Java code. Important syntactic constructs and variable initializations are highlighted. The corresponding code parts between all three programs are visualized via background highlighting. Left: An iterative *BubbleSort* implementation. Middle: An iterative *InsertionSort* implementation. Right: A recursive *InsertionSort* implementation.

Bubble	*Insertion*	recursive
$[4, 7, 2, 1]$	$[4, 7, 2, 1]$	$[4, 7, 2, 1]$
$[4, 2, 7, 1]$	$[4, 2, 7, 1]$	$[4, 2, 7, 1]$
$[4, 2, 1, 7]$	$[2, 4, 7, 1]$	$[2, 4, 7, 1]$
$[2, 4, 1, 7]$	$[2, 4, 1, 7]$	$[2, 4, 1, 7]$
$[2, 1, 4, 7]$	$[2, 1, 4, 7]$	$[2, 1, 4, 7]$
$[1, 2, 4, 7]$	$[1, 2, 4, 7]$	$[1, 2, 4, 7]$

Table 1: The execution traces for the three programs from Figure 1 for the input array $A = [4, 7, 2, 1]$. Only the values for the variable A are shown and intermediate steps that do not manipulate A have been omitted.

that is, for automated classification and analysis of student's computer programs in order to provide helpful, automated feedback.

As an example, consider the programs from Figure 1 again. Their execution traces for the input array $A = [4, 7, 2, 1]$ are shown in Table 1. Despite the apparent syntactic similarity, the implementations of *BubbleSort* and *InsertionSort* do indeed map to different traces, while the iterative and recursive implementation of *InsertionSort* map to the same trace. This indicates that traces have a more direct relationship to the semantics of the underlying program, making them a promising representation for intelligent tutoring systems.

The main contributions of our work are as follows: First, we introduce execution traces with the purpose to capture syntactic as well as semantic aspects of the underlying program (Section 3). Second, we provide an efficient methodology for automatically comparing such traces via edit distances and inferring a measure of similarity for further datamining applications (Section 4). Finally, we evaluate our approach in comparison with the state of the art in syntactic representation in three key challenges for educational data mining: 1.) identifying the student's underlying algorithmic approach (Section 5.2), 2.) identifying erroneous implementations (Section 5.3), and 3.) detecting the location of errors for feedback (Section 5.4). To our knowledge, no data-driven approach exists to date which tackles all three challenges. Syntax-based representations have been successful in identifying the programming strategy [11, 13] but fail in identifying erroneous solutions as well as error locations (as we will show later). On the other hand, test case-based evaluations are very successful in identifying erroneous solutions but treat programs as a black box and thus can make no claims regarding the implemented strategy or the location of the error [17].

2. BACKGROUND AND RELATED WORK

2.1 Tutoring Systems for Computer Programming

In a review of AI-supported tutoring approaches for computer programming, Le and colleagues found six categories of approaches, namely: 1.) displaying examples of programs in order to learn to construct programs of a similar type or modify examples; 2.) simulating the execution of a program in a micro-world and visualizing it to the user; 3.) providing a dialogue environment in order to complete a programming task in an interactive dialogue with the system; 4.) presenting buggy example code in order to learn via program analysis and debugging; 5.) providing feedback to students during development of their program in order to guide them towards a correct solution and detect errors; and finally 6.) providing a collaborative work environment in which students can help each other in developing a program, guided by the system's group model [7]. We note that Le and colleagues do not yet consider recent data-driven approaches, which are mostly feedback-based systems, such as the FIT Java Tutor [2], BOTS [4] and ITAP [16]. Our own approach is targeted mainly at such feedback-based systems working on examples. We analyze the execution trace of a student's program in order to find similar programs for feedback purposes and we intend to locate errors in the student's program to help her correct them. However, our approach also bears similarity to simulation-based approaches as we consider the execution of the program's statements as the main characteristic of a program.

2.2 Representations of Computer Programs for Data-Driven Systems

Most existing data-driven systems for computer programming represent programs as abstract syntax trees, which are subjected to some form of canonalization in order to abstract from mere stylistic differences [15]. Recently, Piech and colleagues have criticized this approach and judged syntax trees not sufficiently discriminative to capture the strong functional consequences of small syntactic changes [14]. Instead, they propose a neural network-based approach to infer a vectorial representation of programs, such that standard machine learning methods can be applied in the resulting Euclidean space. Similar to our approach, Piech and colleagues intend to represent a programs function (or semantics) in opposition to its syntax. However, they focus on a direct mapping between input and output of program segments, while the trace representation provides more procedural (or dynamic) insight into the programs function.

2.3 Edit Distances on Computer Programs

Computing similarities and dissimilarities between computer programs is a crucial step towards data-driven intelligent tutoring system [9]. Edit distances have been particularly prominent in this regard. For example, Rivers and Koedinger used tree edit distances to compute similarities between syntax trees of Python programs to identify adjacent states [16]. Gross and colleagues similarly applied edit distances on syntax trees to infer clusters of computer programs and select the most similar sample solution for feedback [2, 3]. Finally, Paaßen, Mokbel and Hammer have identified the underlying algorithm of sorting programs using machine learning techniques based on alignment distances and adapted the parameters of those alignment distances to yield better classification results [11, 13]. Note that all these approaches rely on alignment distances on program *syntax*, not on execution traces. Striewe and Goedicke applied sequence alignment on execution traces, but did not apply the alignment distances for further datamining purposes [18].

2.4 Classification of Computer Programs

Recently, the value of classification methods for feedback provision in intelligent tutoring systems for computer programming has been recognized. Such machine learning methods enable tutoring systems to infer e.g. the underlying programming strategy of a learner with explicit human labelling only for a small example set [13]. Piech and colleagues report multiplication factors of up to 214, that is, a human tutors annotation for one program permits inference of said annotation for up to 214 other programs [14]. Of course, such approaches rely on a representation of computer programs in a format that can be fed into machine learning methods, such as pairwise similarities and dissimilarities [9, 13] or an explicit vectorial embedding [14]. In this contribution, we employ a classification paradigm to distinguish between different algorithmic approaches, as well as between erroneous and correct solutions.

3. REPRESENTING COMPUTER PROGRAMS VIA EXECUTION TRACES

In general, execution trace recordings can be defined as the "*detection* and *storage* of relevant events during run-time, for later off-line analysis" [6]. More specifically, we consider executions of statements in the program as relevant events, which we characterize by the value of variables of interest after the statement has been executed. This is equivalent to a step-wise execution of the program in a debugger, where we record the state of an interesting variables in each step [17]. As an example, consider traces in Table 1 for the programs in Figure 1.

Only modest technical requirements have to be fulfilled to apply a trace representation. 1.) The programming language has to offer a debugging environment which permits monitoring of a program's execution; 2.) a valid and non-trivial example input for the task has to be available; and 3.) the student's program has to compile and execute without errors on the example input [17]. Thus, the trace representation is more demanding compared to the very flexible syntactic representation of computer programs, but has less prerequisites compared to extensive knowledge engineering. In that sense, the trace representation can be seen as a "middle road" between entirely data-driven approaches and systems based on expert knowledge.

4. COMPARING EXECUTION TRACES

If a student's program is analyzed via test cases, the output is compared with the pre-defined reference value via a simple equality test. However, such a strict equality test is not a viable option for the comparison of execution traces. For example, the traces on the left and the middle in Table 1 are not equal. But they are more similar to each other than to an erroneous program that does not sort the input array at all. Therefore, we require a more flexible measure of similarity or dissimilarity between traces [9].

Similarities and dissimilarities on sequential data can be obtained via *alignment distances* or *edit distances*. The overarching scheme is to extend both input sequences such that there length becomes equal and similar elements of both sequences become aligned. The alignment distance is then defined as the summed cost over all aligned elements [13]. The choice of alignment algorithm depends on the extensions of input sequences that should be permitted. In case of execution traces we intend to abstract from sequence elements that leave the relevant variables unchanged. As an example, consider lines two and three of the program in Figure 1 (left). These two lines could be removed from the program without changing its function, if all expressions of r and l are replaced by their value in the rest of the program. A classic edit distance scheme would punish this with a higher dissimilarity between the shorter and the longer version of the program. Instead, we propose that the same state of the relevant variables may be copied without cost. This corresponds to the *dynamic time warping* dissimilarity D_{DTW} for speech processing, first introduced by Vintsyuk [20]. Given two traces $\bar{x} = (x_1, \ldots, x_M)$ and $\bar{y} = (y_1, \ldots, y_N)$ as well as a dissimilarity measure $d(x_i, y_j)$ between the variable states

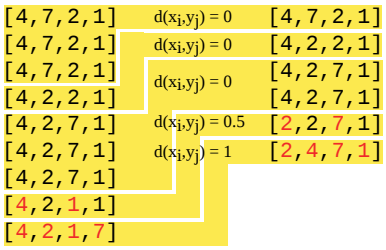


Figure 2: An illustration of the dynamic time warping distance between two traces. Aligned array states are connected by yellow background. Mismatching parts of the aligned variable states are highlighted in red. The dissimilarity between aligned array states is shown in the middle.

x_i and y_j, it is defined recursively as:

$$D_{\mathrm{DTW}}\Big((x_1,\ldots,x_i),(y_1,\ldots,y_j)\Big) := d(x_i,y_j) + \min\Big\{ \quad (1)$$
$$D_{\mathrm{DTW}}\Big((x_1,\ldots,x_{i-1}),(y_1,\ldots,y_{j-1})\Big),$$
$$D_{\mathrm{DTW}}\Big((x_1,\ldots,x_{i-1}),(y_1,\ldots,y_j)\Big),$$
$$D_{\mathrm{DTW}}\Big((x_1,\ldots,x_i),(y_1,\ldots,y_{j-1})\Big)\Big\}$$
$$D_{\mathrm{DTW}}\Big((x_1),(y_1)\Big) := d(x_1,y_1) \quad (2)$$

This can be calculated efficiently in $O(M \cdot N)$ via dynamic programming (D_{DTW} is tabulated for all prefixes of $\bar{x}$ and $\bar{y}$).

An illustration of the dynamic time warping dissimilarity between two example traces is shown in Figure 2. The first three array states of the left trace are just repetitions and thus are aligned with the first array state of the right trace. This occurs again for the fourth to sixth array state of the left trace. Only afterwards the array states differ and lead to a non-zero dissimilarity between both traces. Note that the explicit alignment of array states between two compared traces in dynamic time warping can be retrieved efficiently via backtracing in linear time.

As other edit distances, the dynamic time warping algorithm crucially relies on a dissimilarity measure between variable states. If prior knowledge regarding the interesting variables is available, defining such a measure becomes fairly straightforward (e.g. a Hamming-distance on arrays, just counting the number of unequal entries). In absence of such prior knowledge, defining a dissimilarity on variable states becomes a challenge in itself. One has to infer a semantic matching between the variables in both programs, determine their relevance (as some variables might be less central to the semantic function than others) and then compute the relevance-weighted distance between all matched variables. As a first step in this direction, we propose a simple summary scheme. We build a histogram H_{x_i} in each state x_i that counts the number of variables of each type $t \in \mathcal{T}$, and compare these histograms with a normalized L1 distance:

$$d(x_i,y_j) := \frac{1}{|\mathcal{T}|} \cdot \sum_{t\in\mathcal{T}}^{R} \frac{|H_{x_i}(t) - H_{y_j}(t)|}{|H_{x_i}(t)| + |H_{y_j}(t)|} \quad (3)$$

Note that we consider only types t which occur in both programs at least once.

5. EXPERIMENTS

Our experimental evaluation concerns three key challenges for data-driven intelligent tutoring systems: 1.) Identifying the underlying algorithmic approach, 2.) identifying erroneous programs, and 3.) detecting the location of an error, once a program is identified as erroneous. We compare the performance on these tasks between the trace representation (with dynamic time warping as dissimilarity measure) and the state-of-the-art in terms of syntax representation: syntax-trees with learned edit distance parameters via machine learning techniques [13]. As implementation of the alignment techniques we applied the *TCS Alignment Toolbox* [12].

5.1 Datasets

For our evaluation, we use two benchmark datasets. The *palindrome* data set consists of 48 (correct) programs deciding whether all words in an input sentence are palindromic, using one of eight different programming strategies [9]. We used the histogram-approach to define a dissimilarity between variable states and generated traces using the input sentence "OTTO ANNA MOPS". As this data set does not contain erroneous programs, we only used it for the first experiment.

The second dataset is an extended version of the *sorting* dataset from [11]. It consists of 126 (correct) sorting programs collected from various web sources, each implementing one of six sorting algorithms (35 *BubbleSorts*, 29 *InsertionSorts*, 15 *MergeSorts*, 17 *QuickSorts*, 20 *SelectionSorts* and 10 *ShellSorts*). For each of the programs we created an erroneous counterpart, with one or more *semantic* errors, that is, errors that are neither detected by the compiler nor do they lead to a program crash (e.g. due to an index being out of bounds). Thereby, we focused on errors that are non-trivial to detect for technical systems. As a dissimilarity between variable states we employed a Hamming distance on the array to be sorted. As input we generated a uniform random array of 10 integers in the range $[0, 99]$.

Both datasets are available online at `http://doi.org/10.4119/unibi/2900666` and `http://doi.org/10.4119/unibi/2900684` respectively.

5.2 Classifying Programming Strategies

Our first experiment concerns the identification of the underlying sorting algorithm. We assume that a human expert has already labelled some example programs and want to infer the correct label for some new, unlabelled program. We evaluate the classification accuracy of an 1-nearest neighbor classifier for the syntactic as well as the trace-based representation in a crossvalidation with 6 folds (for the *palindrome* dataset) and 10 folds (for the *sorting* dataset) respectively.

The results are shown in Table 2. For the *palindrome* dataset, the accuracy for the trace representation is more than 10% higher compared to the syntactic representation. Yet, likely due to the small sample size, this difference is not significant (Wilcoxon rank-sum test). In case of the *sorting* data set,

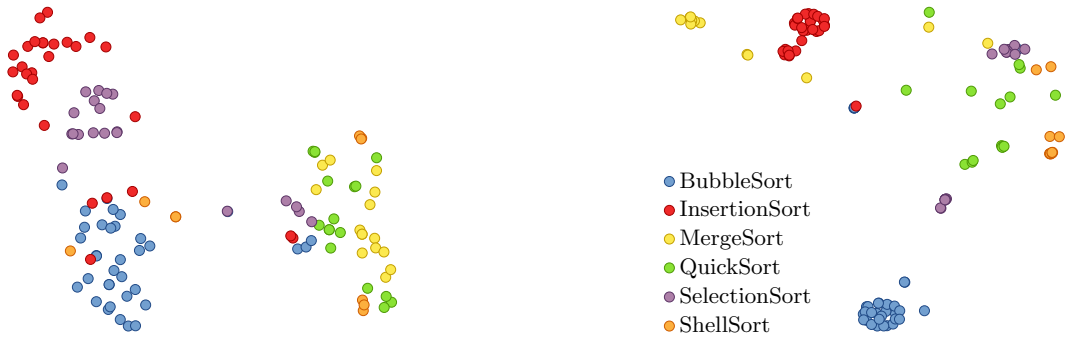


Figure 3: The *sorting* dataset embedded in 2 dimensions via t-stochastic neighborhood embedding (t-SNE) [19]. The sorting algorithms are indicated by color. On the left side, the embedding is shown for adapted, syntactic edit distances [13]. On the right side, we show the embedding for dynamic time warping dissimilarities on traces.

	palindromes		sorting	
method	acc.	std. dev.	acc.	std. dev.
syntax	0.875	0.158	0.812	0.068
traces	0.979	0.051	0.954	0.040

Table 2: The mean classification accuracy and its standard deviation of a 1-nearest neighbor classifier distinguishing six different sorting algorithms. Mean and standard deviation are calculated across 6 (for palindromes) and 10 (for sorting) crossvalidation trials.

we gain an increase in accuracy of more than 14%, which is highly significant ($p < 0.01$, Wilcoxon rank-sum test). This is also reflected in the corresponding dissimilarities. In Figure 3 we show 2-dimensional embeddings of the *sorting* dataset according to syntax-based (left) and trace-based (right) dissimilarities. The trace representation yields more compact clusters corresponding to the correct class label, thereby making classification easier. Interestingly, closer inspection of the misclassified data points for the trace representation revealed that the 1-nearest neighbor classifier correctly identified a *BubbleSort* implementation the programmers had wrongly labelled as an *InsertionSort*.

In order to apply a classification algorithm in praxis, labelled data is required. To reduce human work, one would like to reduce the amount of labelled data necessary. We tested the required amount of labelled data experimentally, by reducing the number of labelled data points (and increasing the number of unlabelled points). The results are displayed in Figure 4. For the *palindrome* data set, only two data points per class are sufficient to achieve good performance. For the *sorting* data set, about 40 labelled programs suffice to achieve a classification accuracy of 90% using the trace representation, while the classification accuracy for the syntactic representation saturates at 80% for about 60 programs.

5.3 Classifying Erroneous Programs

We phrase the identification of erroneous problems as a classification task as well: We assume that a human expert

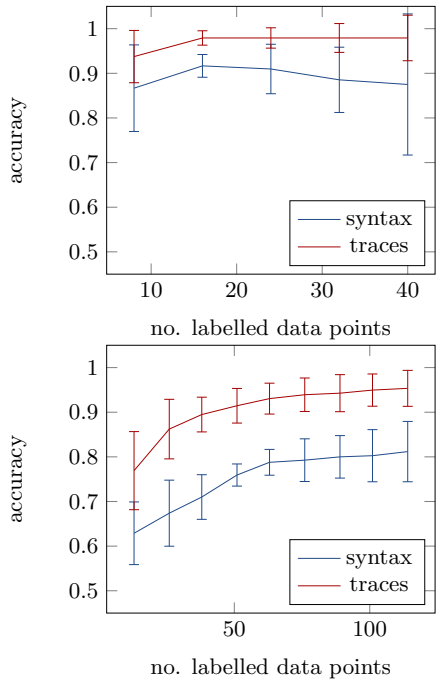


Figure 4: The classification accuracy on the strategy classification task using the syntactic as well as the trace-based data representation if the number of available labelled data points is reduced and the number of unlabelled points is increased. The upper plot displays the result for the *palindromes* dataset, the lower plot for the *sorting* dataset. The errorbars mark the standard deviation across 6 and 10 crossvalidation trials respectively.

method	Accuracy	std. dev.
syntax	0.211	0.107
traces	0.861	0.086

Table 3: The mean classification accuracy and its standard deviation of a 1-nearest neighbor classifier distinguishing erroneous from correct sorting programs. Mean and standard deviation are calculated across 20 crossvalidation trials.

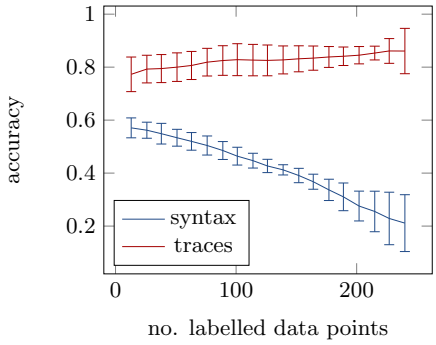


Figure 6: The classification accuracy on the error classification task using the syntactic as well as the trace-based data representation if the number of available labelled data points is reduced and the number of unlabelled points is increased. The errorbars mark the standard deviation across 20 crossvalidation trials.

has labelled a few example programs as correct and erroneous respectively. Then, we want to infer the label for new programs. We evaluate the classification accuracy of an 1-nearest neighbor classifier in a 20-fold crossvalidation.

The results are shown in Table 3. As expected, the syntactic information is not at all sufficient to judge the correctness of a program. The trace-based representation, on the other hand, identifies correct and false solutions in most cases (about 86% accuracy). Again, we can observe the difference between both representation in 2-dimensional embeddings. Figure 5 shows embeddings for the syntactic-based (left) as well as the trace-based dissimilarities (right). While erroneous and correct solutions are almost indistinguishible for the former representation, we observe a much clearer separation of the classes for the latter representation.

We also tested the classification performance if less labelled data is available (see Figure 6). Interestingly, the classification accuracy of the syntactic representation decreases if more labelled data is available. This is likely due to the fact that we created the erroneous programs based on the correct ones, such that the nearest neighbor from a syntactic point of view often was the respective counterpart solution, such that errors get more prevalent if more of such neighbors are available for classification (also refer to Figure 5). Conversely, the trace representation steadily increases in performance and reaches 80% accuracy at about 50 labelled data points.

5.4 Detecting Error Locations

As a final challenge, we try to locate the errors within the erroneous programs. More precisely, the challenge is to identify a set of lines of code in an erroneous program, such that all errors are included, but few other lines are included. Such a set of lines can then be utilized in an intelligent tutoring system. The identified lines can be highlighted such that the student is able to find the error in her program. We apply two strategies based on alignment algorithms, one on the syntactic representation and one on the trace representation.

Syntax-Based Error Detection. We select the nearest correct neighbor and retrieve a syntactic alignment of the erroneous program and the correct program via backtracing. Thereby we obtain the contribution of each line of code in the erroneous program to the overall alignment distance. In order to identify contributing neighbors as well, we apply Gaussian blur to this distribution and then select the line of code with the highest contribution as well as its neighbors, if their contribution is sufficiently high (at least half as high compared to the maximum).

Trace-Based Error Detection. Our trace-based strategy is similar to the syntax-based one. We again select the nearest correct neighbor and retrieve a trace alignment of the erroneous program and the correct program via backtracing. However, we can apply additional domain knowledge. We assume that an erroneous program has the wrong output given the input. The output of the program includes the value of the relevant variables at the end of the trace. Therefore, we can start from the end of the trace alignment and work back until the state of the relevant variables is equal to the state in the correct program. This is the point where the error in the program influences the programs execution negatively. However, it is not sufficient to highlight this particular line of code, because the actual error might be earlier in the code (e.g. a wrongly set index). Therefore, we select not only this line, but the most frequently executed five lines of code until the last change of the relevant variables.

Further, we included three trivial baseline strategies for comparison: 1.) Selecting a line of code at random, 2.) selecting a line of code at random according to its distribution in the trace, and 3.) selecting *all* lines in the program that occured in the trace.

We evaluated all five strategies in a 20 fold crossvalidation. For each erroneous program, we excluded the correct counterpart from the available neighbors in order to make the scenario more realistic.

The results are shown in Table 4. We report the classic pattern recognition measures precision (how many of the selected lines of code contain an error?), recall (how many of the erroneous lines of code have been selected?) and F1-score (harmonic mean of precision and recall). In terms of F1-score, the trace-based error detection method clearly outperforms the syntax-based one ($p < 10^{-4}$, Wilcoxon rank-sum test). Further, as expected, both random baseline meth-

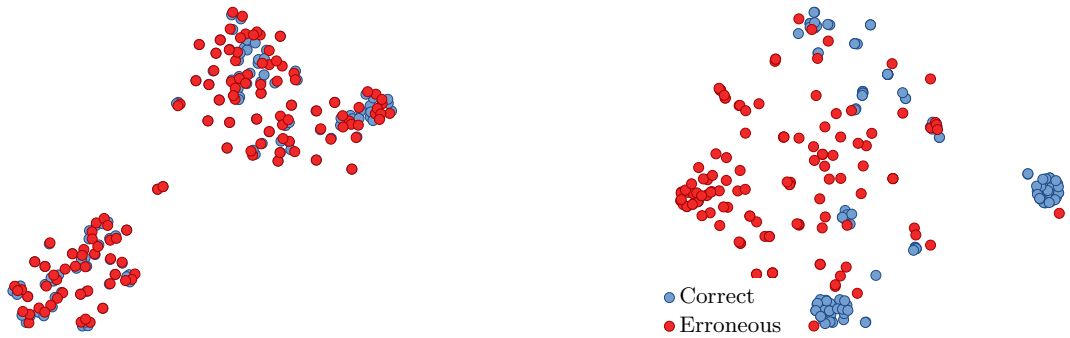


Figure 5: The *sorting* dataset including erroneous solutions embedded in 2 dimensions via t-stochastic neighborhood embedding (t-SNE) [19]. The correctness of each program is indicated by color. On the left side, the embedding is shown for adapted, syntactic edit distances [13]. On the right side, we show the embedding for dynamic time warping dissimilarities on traces.

method	precision	std. dev.	recall	std. dev.	F1 score	std. dev.
traces	0.183	0.071	0.520	0.211	0.269	0.104
syntax	0.103	0.086	0.134	0.100	0.115	0.091
traces_random	0.157	0.122	0.119	0.098	0.134	0.107
syntax_random	0.121	0.116	0.095	0.095	0.105	0.103
traces_all	0.103	0.022	0.976	0.050	0.186	0.037

Table 4: The mean classification accuracy and its standard deviation of a 1-nearest neighbor classifier distinguishing erroneous from correct sorting programs. Mean and standard deviation are calculated across 20 crossvalidation trials.

ods seldomly select an erroneous line, thereby limiting the recall. However, selecting all lines of code occuring in a trace provides a strong baseline to compete with ($F1 = 0.186$). Still, the trace-based error location method performs significantly better ($p < 0.01$, Wilcoxon rank-sum test).

6. DISCUSSION

In this contribution we introduced an alternative representation of computer programs for classification and error detection in intelligent tutoring systems (ITSs), namely execution traces. On two example data sets we have demonstrated that this representation can improve upon state-of-the-art syntax-based representation in terms of strategy classification, error classification and error detection. In a full-blown ITS for computer programming, the trace representation can thus be applied to help students in solving programming tasks. As soon as a student has managed to reach a working state (without syntax errors and program crashes) we can generate a trace and compare it with the traces of different programs. The resulting (dis-)similarity measure can be used to identify possible partners for peer-review and peer-tutoring by matching students that apply the same approach in their solution attempt. Further, the trace representation can be applied to identify erroneous programs, enabling an ITS to detect whether a student has finished a task or still needs to continue. Further, as not only the end result is checked but the whole execution, the trace representation can be utilized for detecting unusual or deceptive solutions that are geared towards the test cases without actually implementing the desired function. Finally, if an error is still present in a student's program but the error is not obvious, the trace representation may help to identify and highlight the location of the error in the program code, thereby providing scaffolding to students that get stuck in searching for their error.

Overall, the trace representation appears to be highly useful for data-driven ITSs on computer programming. However, important challenges remain. If no a priori knowledge regarding the relevant variables in the program is available, computing a dissimilarity on variable states is not trivial. We have suggested a first attempt using a histogram of variable types. This representation, however, disregards the content of variables and thus is likely not sufficiently powerful in many applications where differences in variable values are important markers of program semantics. A solution might be to match variables probabilistically according to the alignment distance a certain matching produces. This is an interesting direction to pursue in further research.

Finally, we note that the trace representation does not have to be the sole source of information for an ITS. A syntactic representation is necessary when a program does not yet compile or crashes and wherever the high level of abstraction applied by a program trace is not helpful (e.g. when teaching certain syntactic constructs). Fusing the strengths of both representations is likely to lead to the best learning outcomes for students.

7. ACKNOWLEDGMENTS

Funding by the DFG under grant number HA 2719/6-2 and the CITEC center of excellence (EXC 277) is gratefully acknowledged.

8. REFERENCES

[1] J. R. Anderson and E. Skwarecki. The automated tutoring of introductory computer programming. *Commun. ACM*, 29(9):842–849, Sept. 1986.

[2] S. Gross, B. Mokbel, B. Paaßen, B. Hammer, and N. Pinkwart. Example-based feedback provision using structured solution spaces. *International Journal of Learning Technology*, 9(3):248–280, Nov. 2014.

[3] S. Gross and N. Pinkwart. How do learners behave in help-seeking when given a choice? In C. Conati, N. Heffernan, A. Mitrovic, and M. F. Verdejo, editors, *Artificial Intelligence in Education*, volume 9112 of *Lecture Notes in Computer Science*, pages 600–603. Springer International Publishing, 2015.

[4] A. Hicks, Y. Dong, R. Zhi, V. Catete, and T. Barnes. BOTS: selecting next-steps from player traces in a puzzle game. In *Workshops Proceedings of EDM 2015 8th International Conference on Educational Data Mining, EDM 2015, Madrid, Spain, June 26-29, 2015.*, 2015.

[5] K. R. Koedinger, E. Brunskill, R. S. Baker, E. A. McLaughlin, and J. Stamper. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, 34(3):27–41, 2013.

[6] J. Kraft, A. Wall, and H. Kienle. Trace Recording for Embedded Systems: Lessons Learned from Five Industrial Projects. In H. Barringer, Y. Falcone, B. Finkbeiner, K. Havelund, I. Lee, G. Pace, G. Roşu, O. Sokolsky, and N. Tillmann, editors, *Runtime Verification: First International Conference, RV 2010, St. Julians, Malta, November 1-4, 2010. Proceedings*, pages 315–329. Springer Berlin Heidelberg, 2010.

[7] N. T. Le, S. Strickroth, S. Gross, and N. Pinkwart. A review of ai-supported tutoring approaches for learning programming. In N. T. Nguyen, T. Do, and H. A. Thi, editors, *Advanced Computational Methods for Knowledge Engineering - Proceedings of the 1st International Conference on Computer Science, Applied Mathematics and Applications (ICCSAMA)*, number 479 in Studies in Computational Intelligence, pages 267–279, Berlin, Germany, 2013. Springer Verlag.

[8] C. Lynch, K. D. Ashley, N. Pinkwart, and V. Aleven. Concepts, structures, and goals: Redefining ill-definedness. *International Journal of Artificial Intelligence in Education*, 19(3):253–266, 2009.

[9] B. Mokbel, S. Gross, B. Paaßen, N. Pinkwart, and B. Hammer. Domain-Independent Proximity Measures in Intelligent Tutoring Systems. In S. K. D'Mello, R. A. Calvo, and A. Olney, editors, *Proceedings of the 6th International Conference on Educational Data Mining (EDM)*, 2013.

[10] T. Murray, S. Blessing, and S. Ainsworth. *Authoring tools for advanced technology learning environments: Toward cost-effective adaptive, interactive and intelligent educational software*. Springer, 2003.

[11] B. Paaßen, B. Mokbel, and B. Hammer. Adaptive structure metrics for automated feedback provision in java programming. In M. Verleysen, editor, *23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pages 307–312, 2015.

[12] B. Paaßen, B. Mokbel, and B. Hammer. A toolbox for adaptive sequence dissimilarity measures for intelligent tutoring systems. Proceedings of the 8th International Conference on Educational Data Mining, pages 632–632. International Educational Datamining Society, 2015.

[13] B. Paaßen, B. Mokbel, and B. Hammer. Adaptive structure metrics for automated feedback provision in intelligent tutoring systems. *Neurocomputing*, 192, 2016.

[14] C. Piech, J. Huang, A. Nguyen, M. Phulsuksombati, M. Sahami, and L. Guibas. Learning program embeddings to propagate feedback on student code. In *Proceedings of the 32nd International Conference on Machine Learning*, International Conference on Machine Learning, pages 1093–1102, 2015.

[15] K. Rivers and K. R. Koedinger. A canonicalizing model for building programming tutors. In S. A. Cerri, W. J. Clancey, G. Papadourakis, and K. Panourgia, editors, *Intelligent Tutoring Systems: 11th International Conference, ITS 2012, Chania, Crete, Greece, June 14-18, 2012. Proceedings*, pages 591–593. Springer Berlin Heidelberg, 2012.

[16] K. Rivers and K. R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *International Journal of Artificial Intelligence in Education*, pages 1–28, 2015.

[17] M. Striewe and M. Goedicke. Using run time traces in automated programming tutoring. In *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education*, ITiCSE '11, pages 303–307, New York, NY, USA, 2011. ACM.

[18] M. Striewe and M. Goedicke. Trace alignment for automated tutoring. In *Computer Assisted Assessment Conference*, 2013.

[19] L. van der Maaten and G. E. Hinton. Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[20] T. Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4(1):52–57, 1968.

Generating Data-driven Hints for Open-ended Programming

Thomas W. Price
North Carolina State Univ.
890 Oval Drive
Raleigh, NC 27695
twprice@ncsu.edu

Yihuan Dong
North Carolina State Univ.
890 Oval Drive
Raleigh, NC 27695
ydong2@ncsu.edu

Tiffany Barnes
North Carolina State Univ.
890 Oval Drive
Raleigh, NC 27695
tmbarnes@ncsu.edu

ABSTRACT

Intelligent Tutoring Systems (ITSs) have shown success in the domain of programming, in part by providing customized hints and feedback to students. However, many popular novice programming environments still lack these intelligent features. This is due in part to their use of open-ended programming assignments, which are difficult to support with existing hint generation techniques. In this paper, we present a new data-driven algorithm, based on the Hint Factory, to generate hints for these open-ended assignments. We evaluate our algorithm on historical student data and show that it can provide hints that successfully lead students to solutions from any state, help students achieve assignment objectives, and align with the student's future solution.

1. INTRODUCTION AND BACKGROUND

Intelligent Tutoring Systems (ITS) have shown much promise in the domain of computer programming [3, 14, 16, 22], with studies arguing that students using an ITS perform as much as two standard deviations higher than those who receive conventional instruction [3]. A key feature of any ITS is the ability to give students context-sensitive feedback during problem solving, often in the form of hints. In the domain of programming, this feedback has been shown to improve students' performance, both inside the tutor and on subsequent assessment [4].

Despite positive empirical evaluations, these specialized ITSs are not generally used in introductory programming classes. In particular, new introductory Computer Science (CS) curricula, such as CS Principles[1] and Exploring CS[2] are turning to programming environments designed specifically for novices, such as Scratch [19], Snap [8] and Alice [5], which engage students in creating open-ended projects, such as games, stories and simulations [25]. These environments have features specifically designed for novices, such as drag-and-drop, block-based interfaces that improve student performance by minimizing the challenges of syntax [18]. They offer improved outcomes over traditional instruction, such as increased retention [11] and improved test scores [5].

Unfortunately, aside from some preliminary research [2], little effort has been made to bring the intelligent features of ITSs to these novice programming environments. This is due in part to the large investment of time required by domain experts to create these systems, which has been estimated as high as 300 hours to create one hour of intelligent content [12]. Further, the use of open-ended programming assignments, which makes these environments so appealing to students and teachers, also serves as a major barrier to providing intelligent, adaptive feedback. These assignments often have multiple, loosely ordered objectives, which cannot be assessed automatically, making it difficult to apply automatic hint generation techniques that rely on test cases (e.g. [15, 22, 23]).

Data-driven tutors have the potential to overcome these barriers. The Hint Factory is an algorithm that has been used to generate data-driven hints from historical student data, originally in the domain of logic proofs [1]. The Hint Factory is like a recommender system that uses student data as a basis for automatic hint generation, making it easy to scale up without additional expert involvement. The Hint Factory has been successfully adapted to the domain of programming in a variety of ways [14, 9, 22]. However, data-driven hints have not been evaluated on open-ended assignments in novice programming environments, and may not be well equipped to handle them [17].

In this paper we present an extension of the Hint Factory specifically designed to provide hints to students working on open-ended programming assignments. The algorithm is fully data-driven, requiring no reference solution or test cases, and presents hints that represent real student actions. It is designed to be programming language and system agnostic, with the intention of making it applicable to a variety of novice programming environments. We evaluate this algorithm on historical student data from an open-ended assignment in a novice programming environment, and show that it is capable of providing hints that successfully lead students to solutions from any state, help students achieve assignment objectives, and align with the student's future solution.

1.1 The Hint Factory

The Hint Factory [24] is an algorithm for generating next-step hints for students working on multi-step problems. It operates on a data-structure called an interaction network [6], which is built from log data of the interactions between students and a learning environment for a given problem. The interaction network is a directed graph, where each vertex represents a state of the problem. In programming a state corresponds to a snapshot of the student's current work (code). States are connected by edges, which represent student actions, such as adding, editing or deleting code, which transform one state into another. Each student attempt is traced from a start state to its final state and is added to the interaction network. If this final state is a correct solution, we label it as a goal state. By combining all students' attempts into a single network and weighting edges

[1] www.csprinciples.org
[2] www.exploringcs.org

with the number of attempts that passed through them, the interaction network forms a compact representation of student problem solving strategies for a given problem.

The Hint Factory uses the interaction network for a given problem to generate hints for new students working on that problem. When a student requests a hint, the algorithm matches that student to an existing state in the network and then calculates the best path from that state to a goal state. The Hint Factory uses a Markov Decision Process (MDP) to calculate this solution path [1], but other techniques can also be used, which are more effective in some contexts [16]. Once a solution path is calculated, it is typically used to provide a next-step hint, which points the student towards the next state in the solution path. The exact method of suggesting this state as a hint is system-dependent.

1.2 Hint Generation in Programming

The domain of computer programming presents a serious challenge for automatic hint generation, especially for data-driven systems. Even for simple programming problems, the space of possible solutions is quite large, often infinite, and there may be little overlap among student solutions [17, 20]. Many automated hint generation algorithms search through this space, attempting to transform a student's current program into a solution state using some sort of program generation or synthesis [10, 15, 22, 23, 26]. These techniques require an expert-supplied reference solution and/or set of test cases to ensure that generated programs are correct. To facilitate this transformation, algorithms often represent a student's program using an Abstract Syntax Tree (AST), a directed, rooted tree where each node represents a program element, such as a function call, control structure or variable, and the hierarchy of the tree represents how these elements are nested together.

Zimmerman and Rupakheti [26] use a pq-Gram tree edit distance algorithm to match a student's program to its closest counterpart in a database of target solutions, as well as to identify the set of insertions, deletions and relabelings that will directly transform the student's AST into this solution. Rather relying on a fixed set of solutions, Singh et al. [23] use program synthesis to generate a new solution from the student's current program. They do so using an expert-provided Error Model, which defines a set of potential transformations to a student's code for a given problem. Other techniques are data-driven like the Hint Factory, using previous student solutions to provide hints. Perelman et al. [15] also employ program synthesis to search for a solution program, using a Domain-Specific language (DSL) to define possible program transformations; however, they show that this DSL can be automatically generated from previous student solutions. Our approach also works to transform a student's program into a solution, but rather than using an automated technique like program synthesis, we use edits from actual students. Lazar and Bratko [10] employ a similar approach, applying single-line edits observed in previous student work to transform a student's program into a solution; however, their technique requires a set of test cases to evaluate generated programs, and ours does not.

The Hint Factory has also been adapted to the domain of programming, with modifications to address the large state space and lack of overlap among student solutions. Rivers and Koedinger [22] extend the Hint Factory using a strategy called path construction to generate a path from a student's current state to a previously observed goal state, rather than relying on observed student paths. They compute a change vector of all edits needed to transform the student's current state into the goal state and test to see if any closer solutions are discovered along the way. Peddycord III et al. [14] applied the Hint Factory to a programming game called BOTS, but rather than representing a student's state using an AST (a *codestate*), they used the state of the game world after running the student's program (a *worldstate*). The authors found considerably more overlap among worldstates than codestates, allowing more hints to be generated; however, these hints may be more challenging to apply. Fossati et al. [7] used a similar approach to the Hint Factory to generate both reactive and proactive data-driven feedback in the iList linked list tutor. They found that with this feedback, iList produced equivalent learning gains to a human tutor.

Most methods for hint generation benefit from overlap among student programs. This overlap can be increased through canonicalization, which standardizes the *syntax* of programs, while maintaining their *semantic* meaning. For example, the expression $a > b$ can be rewritten $b < a$ without changing its meaning. Rivers and Koedinger [20] present a comprehensive technique for canonicalization, which standardizes programs in a variety of ways, such as normalizing arithmetic and boolean operators, removing unreachable and unused code and inlining helper functions. Jin et al. [9] take a different approach, representing a student's program as a Linkage Graph, where each vertex is a code statement, and each directed edge represents an ordering dependency. This removes some semantically unimportant ordering information from the program, allowing for more overlap.

2. THE CTD ALGORITHM

In this section we present the Contextual Tree Decomposition (CTD) algorithm for hint generation, our extension of the Hint Factory to the domain of open-ended programming problems. Existing hint generation techniques are effective on traditional programming assignments with single objectives that are easily assessed with test cases. Open-ended assignments, by contrast, may have multiple, loosely ordered objectives that do not lend themselves to automated assessment, as they often deal with user interaction or graphical output. As such, we cannot rely on the program generation techniques discussed in Section 1.2 to create hints. Instead, we take a fully data-driven approach, using student data, rather than automated search, to construct a path to a goal state. Not only does this approach make hint generation feasible for open-ended assignments, it also has the advantage of presenting hints that correspond to real student actions, which should be understandable to other students.

2.1 An Example Assignment

To illustrate the CTD algorithm, we will use an assignment called the "Guessing Game" as a running example throughout this section. In the Guessing Game, students are asked to create a program that stores a random number and then repeatedly asks the player to guess it until they are correct, informing them if they have guessed too high or too low. To begin, the game should welcome the player and greet them by name. The assignment requires the use of loops, conditionals, variables and various arithmetic operators. A common implementation of the Guessing Game is presented in Figure 1.

Note that this is one of many possible solutions to the problem. For example, we could use three `if` statements, rather than an `if/else` block. Now consider a student, Alice, working on

```
GuessingGame:
   Say( "Welcome to the Guessing Game!" )
   answer ← Ask( "What is your name?" )
   Say( Join( "Hello ", answer ) )
   number ← Random( 1, 10 )
   doUntil ( answer == number ):
      answer ← Ask( "Guess a number" )
      if ( answer == number ):
         Say( "Correct!" )
      else:
         if ( answer > number ):
            Say( "Too high!" )
         if ( answer < number ):
            Say( "Too low!" )
```

Figure 1: An example solution to the Guessing Game assignment.

```
GuessingGame:
   number ← 8
   Say( "Welcome!" )
   answer ← Ask( "Who's playing?" )
   Say( Join("Hi ", answer ) )
   doUntil ( answer == Random( 1, 10 ) ):
      answer ← Ask( "Guess a number" )
```

Figure 2: An example of a partial, flawed solution attempt from a student, Alice.

the Guessing Game with code presented in Figure 2. Alice has added the first few lines of code in a different (but correct) order; however, she does not understand how to store and use the random number for the guessing game. A hint could demonstrate the correct behavior for her.

2.2 Generating Hints

In the CTD algorithm, as in previous work, we represent a student's state using an AST. Borrowing from Rivers and Koedinger's work [20], we also use basic canonicalization to increase overlap among ASTs. In our ASTs, we use a single label for all variables (`var`) and for all literals (`literal`). The arguments of commutative operators (e.g. `==`, `+`, `*`) are given a fixed ordering, and we rewrite any greater than expression $x > y$ as a less than expression $y < x$. A canonicalized AST for the code presented in Figure 1 is shown in Figure 3.

Most data-driven hint generation algorithms attempt to answer the question, "Given a student's current state, what should their next state be?" Rather than trying to answer this question for a student's entire program, we try to answer it for the children of each node of a student's AST. For example, if Alice were to request a hint, we might tell her to assign a different value to `number`, compare different values using `==` or add code to the body of `doUntil`. By breaking the student's program down into a set of smaller pieces, we can more easily match it to the programs of previous students, as suggested in previous work [10, 21].

To generate hints from student data, we build a *set* of *contextual* interaction networks (CINs), which each model how students edit a subsection of the program over time. We build one CIN for

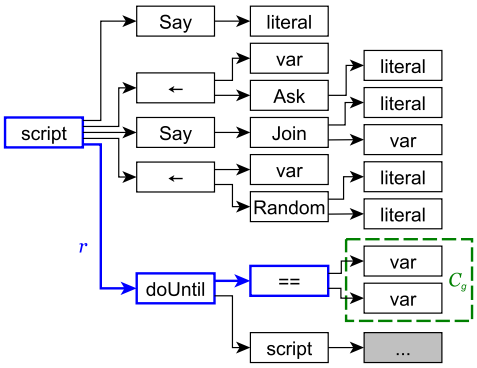


Figure 3: A partial AST for the code shown in Figure 1. A root path r is outlined in bold blue, with its current state (C_g) in dashed green.

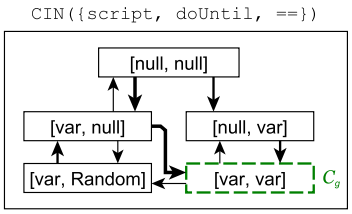


Figure 4: The contextual interaction network CIN({`script, doUntil, ==`}) with goal state C_g. Edge thickness represents transition frequencies.

each unique *root path* observed in all students' ASTs (including ASTs from intermediate code snapshots). A root path (RP) for a node n in an AST is the path from the root node to n. Figure 3 highlights an example RP for the `(==)` node: `{script, doUntil, ==}`. Some nodes have the same root path, such as the two `(Say)` nodes, which have the RP `{script, Say}`. Each RP r corresponds to a unique CIN, denoted CIN(r), which functions just as the interaction networks described in Section 1.1. However, CIN(r) only models changes to the *immediate children* of the last node in r. For example, CIN(`{script, doUntil, ==}`), shown in Figure 4, models changes to the children (operands) of the `(==)` node. Each state in CIN(r) is a list of the children of the last node in r, and each edge represents an edit to those children. Figure 3 highlights C_g, the list of children of the `(==)` node, which corresponds to a state in the CIN shown in Figure 4. Because the AST shown in Figure 3 is a correct solution, C_g is a goal state in CIN(`{script, doUntil, ==}`). Given that Alice's current state in this CIN is `[var, Random]`, to get to the goal state C_g we would recommend that she delete her `(Random)` node and then replace it with a `(var)` node.

The procedure for building the CINs from previous data is shown in Algorithm 1. We represent a student's work as a sequence of ASTs, T, where each tree t_i in the sequence is a snapshot of the student's work at time i, and the last tree represents the submitted solution attempt. For each sequential pair of trees, t_i and t_{i+1}, we find all pairs of AST nodes (n_i,n_{i+1}) that represent

the same code element in both trees, and therefore have the same RP r. We examine the lists of child nodes C_i of n_i and C_{i+1} of n_{i+1} in their respective ASTs. If C_i and C_{i+1} are different, we add the states C_i and C_{i+1} to CIN(r) (if they do not already exist) and add an edge from C_i to C_{i+1}. This edge represents how the student has edited the code in this part of the AST from time i to time i+1. Algorithm 1 runs in $O(|T||t_m|^2)$ time for a given student, where $|T|$ is the number of ASTs recorded for that student and $|t_m|$ is size of the largest recorded AST.

Algorithm 1 Add a Student to the CINs

Require: A sequence of student ASTs T
Ensure: Student data has been added to relevant CINs
  for all t_i,$t_{i+1} \in T$ **do**
    for all $(n_i,n_{i+1}) \in$ MatchingNodes(t_i, t_{i+1}) **do**
      $r \leftarrow$ RootPath(n_i)
      $C_i \leftarrow$ Children(n_i)
      $C_{i+1} \leftarrow$ Children(n_{i+1})
      if $C_i \neq C_{i+1}$ **then**
        AddEdge(CIN(r), C_i, C_{i+1})
      end if
    end for
  end for

Once we have added student data to the CINs, we can generate hints for new students, as shown in Algorithm 2. Because we now have many CINs, rather than a single interaction network, we also generate a *set* of hints. For each node n in a student's current AST, we calculate its root path r and find CIN(r). The student's current state in CIN(r) is C_0, the list of children of n. We then use the Hint Factory algorithm [1] to generate a hint using the interaction network CIN(r) and the student's current state in the network C_0. This hint will recommend a new set of children C_1 for n, which we can then display as a suggestion to the student. Note that if C_0 is already a goal state the Hint Factory will recommend that the student stay in that state, in which case $C_0 = C_1$ and we present no hint for n. Algorithm 2 runs in $O(|t|^2+|t||S_m|^2)$ time [3], where $|t|$ is the size of the student's AST and $|S_m|$ is the number of states in the largest CIN(r). In practice, $|S_m|$ remains small, as a given CIN models changes to only a small part of a student's code.

Algorithm 2 Get Hints

Require: The student's current AST t
Ensure: H is a set of node-hint pairs
  $H \leftarrow \{\}$
  for all $n \in$ Nodes(t) **do**
    $r \leftarrow$ RootPath(n)
    $C_0 \leftarrow$ Children(n)
    $C_1 \leftarrow$ HintFactoryHint(CIN(r), C_0)
    $H \leftarrow H \cup \{(n, C_1)\}$
  end for

A classic challenge for the Hint Factory is how to provide hints to states with no exact matches in the interaction network. CINs break a program down into smaller parts to provide more opportunities for matches, but this does not guarantee a match. If no exact match is found for a state C_0, we find the closest state to C_0 in the CIN and use it as a next-step hint. Because CIN states are lists of children, we can use a simple edit distance to determine the closest state. If the distance between the current state and its closest pair in the CIN is beyond a certain threshold (e.g. 3 edits), we assume the student is doing something unknown, and we do not provide a hint for that state.

[3]This assumes we use a constant bound on the number of iterations allowed during the Hint Factory's value iteration.

2.3 Goal States

In order to run on an interaction network, the Hint Factory requires a reward function $R(s)$, which is used by the MDP to assign a reward to each state in the network [1]. Traditionally, this value has been some large number (e.g. 100) for goal states and 0 otherwise. However, in many open-ended programming problems, we cannot automatically determine whether or not a given program state satisfies the goal of the assignment. A simple solution is to assign a reward value to each state proportional to the number of students who submitted a program in that state. We accomplish a similar effect with CINs by finding each node n and corresponding RP r in each student's submitted AST and marking the list of children of n as a goal in CIN(r).

One challenge with CINs is that two different parts of a program may correspond to the same CIN. For example, recall that the two `Say` statements in Figure 3 have the same RP, and thus the same CIN, but ideally these two nodes should end up in two different goal states. The first should end up with children `[literal]`, while the second should have children `[Join]`. Both of these states will be marked as goals in the shared CIN, so how can the algorithm determine when one goal should be chosen over the other?

To address this, each time a node's children are marked as a goal state in a CIN, we also store that node's *context*. This context helps identify when a particular goal state might be applicable. We define a node's context using two lists, consisting of its left and right siblings in the AST. For example, in Figure 3, the first `Say` node has a context `{[], [←, Say, ←, doUntil]}`, while the second node has a context `{[Say, ←], [←, doUntil]}`. Rather than giving goal states a fixed reward value, we determine this value individually for each hint request. For each previous student attempt that finished in a given goal state, we increase the reward for that state by a value inversely proportional to the distance between the previous attempt's context and the current attempt's context. Again, because the contexts consist of lists, a simple edit distance can serve as a distance metric.

2.4 Smoothing Hints

The Hint Factory is typically used to generate a next-step hint, which suggests the next state a student should achieve. The advantage of the Hint Factory is that this action has been done by a previous student, and is therefore likely to seem reasonable to the current student. However, sometimes the path that a real student takes to a solution can be circuitous. Students often add code that they later delete, or add code in one place and later move it to another. In these cases we use the entire *solution path* generated by the MDP, rather than a single state, to make suggestions that will not be contradicted by future hints. We call this process "smoothing", since it will make hints appear more consistent.

We use Algorithm 3 to generate hints which follow real students' paths, while avoiding unnecessary or contradictory edits. We first calculate a full solution path from the student's current state to a goal state using the Hint Factory on the CIN, as described

earlier. Recall that each state in this path is a list of child nodes in the AST. We first reorder nodes in the student's current state to match the goal state ordering. We then insert any new nodes from the next state in the solution path (like set union) and reorder the nodes again to match the goal state. Finally, we remove any nodes that are not in the goal state (like set intersection). If the resulting state is not different than the student's current state, we repeat the process with the next state in the solution path. Using this "smoothing" process helps us avoid giving hints that add code that will later need to be moved or deleted.

Algorithm 3 Get Smoothed Hint

Require: The MDP of a CIN and student's *state*
Ensure: *hint* is a smoothed hint for the student
  $path \leftarrow$ GetSolutionPath(*state*, MDP)
  $goal \leftarrow$ Last(*path*)
  $hint \leftarrow state$
  $hint \leftarrow$ Reorder(*hint*, *goal*)
  for all $s_i \in path$ **do**
    $hint \leftarrow hint \cup s_i$
    $hint \leftarrow$ Reorder(*hint*, *goal*)
    $hint \leftarrow hint \cap goal$
    if $hint \neq state$ **then**
      return
    end if
  end for

3. METHODS

We evaluated the efficacy of the CTD algorithm using data from real students working on the Guessing Game assignment described in 2.1. Data was collected from an introductory undergraduate computing course for non-CS majors during the Fall of 2015, which had approximately 80 students. The first half of the course focused on learning the Snap programming language through a curriculum based on the Beauty and Joy of Computing (BJC) [8]. Snap is a visual programming environment that allows users to create media-rich, interactive programs by dragging blocks of code together to form scripts. Students worked on the Guessing Game assignment during class for approximately one hour, with a teaching assistant (TA) available to assist them and the ability to discuss the assignment with nearby students. We collected trace log data of all student interactions with the programming environment. After each edit to a student's program, the complete program state (a snapshot) was recorded. For the "Guessing Game" assignment, we collected 51 attempts, consisting of 8666 total code snapshots.

Each of the final submissions was graded by two independent graders. The graders used a rubric consisting of nine assignment objectives, such as welcoming the player by name, storing a secret number, and repeatedly asking the player for guesses. The graders had an initial agreement of 94.5%, with Cohen's $\kappa = 0.544$, and after clarifying objective criteria and independently re-grading this rose to 98.1%, with Cohen's $\kappa = 0.856$. Any remaining disagreements were discussed to create final grades for each assignment. The students achieved on average 92.8% of objectives, with all students getting at least 4 out of 9. The high grades can be attributed in part to the presence of TAs, who helped struggling students to complete the assignment. Using the same criteria, an automatic grading program was created, which manually checked code structure for objective completion. The automatic grader was tested on the manually graded data, achieving 100% accuracy on 7 of 9 objectives. On each of the remaining two objectives, it incorrectly marked two submissions as failing since they used atypical approaches. Note that this grader was used in our evaluation but not for hint generation.

We generated and evaluated hints for each code snapshot of each student in our dataset (n=8666), giving us a clear view of hint performance across students and time. We evaluated the hints using a number of criteria, detailed in Section 4. Because Snap lends itself to a "tinkering" approach, code snapshots often contain many extra scripts that students keep in their workspace for later use. Since the Guessing Game uses only one script, these extra scripts do not reflect the student's primary work, and it would not make sense to evaluate hints for them. Therefore, in our analyses we considered only the largest script in a snapshot.

3.1 Hint Policies

To better evaluate the CTD algorithm, we generated hints using four hint policies:

1. **CTD All** (CA): Hints are generated using CTD on all student data (n=51).
2. **CTD Exemplar** (CE): Hints are generated using CTD on data from only exemplar students, whose final submissions achieved all assignment objectives (n=32).
3. **Direct Expert** (DE): Hints modify a student's program directly towards an expert solution using a single node insertion, deletion or relabeling.
4. **Direct Student** (DS): Hints modify a student's program directly towards their own submitted solution, using a single node insertion, deletion or relabeling.

The CA and CE policies both use the CTD algorithm, and comparing them allows us to explore the effect of including students with incorrect final solutions on the algorithm's output. The DE and DS policies both generate hints using a technique outlined by Zimmerman et al. [26], which identifies the node insertions, deletions and relabelings required to transform one AST into another. Each of these modifications is treated as a hint. The DE policy targets a single expert solution, while the DS policy targets the student's own future final solution, and could not actually be implemented on real-time data. In many ways, the DS policy represents an ideal hint policy for students who achieve a correct final solution (which the majority of our sample did), as it perfectly anticipates their solution strategies.

All policies generate a set of hints, where each hint represents a small modification to a student's program. When generating hints for a given student using CTD, we did not include that student in the dataset used to build the CINs, similar to leave-one-out cross validation, since that student's future data could not be used in a real-time setting.

4. EVALUATION AND DISCUSSION

Our evaluation of CTD focused on the following research questions:

RQ1 Can CTD successfully lead students to a solution regardless of their current program?

RQ2 Can CTD hints help students complete objectives?

RQ3 How consistent are CTD hints with student actions?

RQ1 asks whether CTD solves the challenge of generating hints for an open-ended problem where there is little exact overlap among student solution paths. RQ2 investigates whether these hints are good in that they leads student to complete assignment objectives. Lastly, RQ3 asks whether the hints that CTD provides point students in what might be perceived as a reasonable direction, so students will be inclined to use them. Our evaluations for RQ2 and RQ3 compared the CA and CE policies with the baseline DE and DS policies discussed in Section 3.1.

4.1 Providing Hints

RQ1 asks whether CTD can successfully generate hints for solution attempts regardless of how much overlap they have with other attempts in our dataset. Therefore, we first examined how much overlap there was in our dataset. We recorded 8666 snapshots from 51 students; however, many students produced duplicate snapshots, for example by adding and then removing an element of code. If we do not count duplicate snapshots from the same student, we are left with 5103 snapshots. If we also ignore all but the largest script from these snapshots (as is done in our analyses), there are 3181 non-duplicate snapshots. Of these, 2714 (85%) were unique after canonicalization, meaning they showed up in only one student's data. In addition, 47 of 51 students had unique final solution ASTs. We conclude that the state space is quite sparse, with little overlap among student solution paths.

We evaluated hints from the CA and CE policies to determine if they could get students to a solution despite this sparsity. To align student attempts over time, and to balance our sample evenly across students, we took 50 snapshots from each student, spaced evenly throughout their progression, and called these "slices." For each student, we generated a *hint chain* from each of these 50 snapshots to a final solution. A hint chain is the sequence of program states that would result if the students followed sequential "top-level" CTD hints from a given snapshot to program completion. The top-level hint is that which comes from the CIN(r) with the shortest RP r.

Both CA and CE policies were able to generate successful hint chains for every slice, meaning the hint policies always had a hint to provide and there were no hint cycles. Figure 5 shows the average hint chain length for each slice. Both policies showed a steady, near-linear decline in hint chain length over time. This supports the notion that CTD makes good use of the student's existing work. On average, students took 175.8 steps to complete the assignment, so both policies are more efficient than the student until slice 46/50. As students converge on their own solutions, however, the hints chains become less efficient, as they often lead students to alternative solutions.

To understand the quality of solutions created using hint chains, we evaluated the final solutions generated by the hint chains at each slice using the automatic grader discussed in Section 3. The CA policy solutions received grades averaging from 89.5-93.0% across slices, while the CE policy averaged 98.5-100% across slices. Upon closer inspection, we found that all imperfect CA solutions were identical and satisfied 8 of 9 objectives (88.9%), and all correct CA solutions were identical as well. The one objective missed by the imperfect CA solution was also missed by 12 of 51 students (23.5%), indicating that a frequent enough mistake in student data will be reflected in CTD hints. The CE policy produced 3 unique, correct solutions and 2 unique, incorrect solutions, which both satisfied the same 8 out of 9

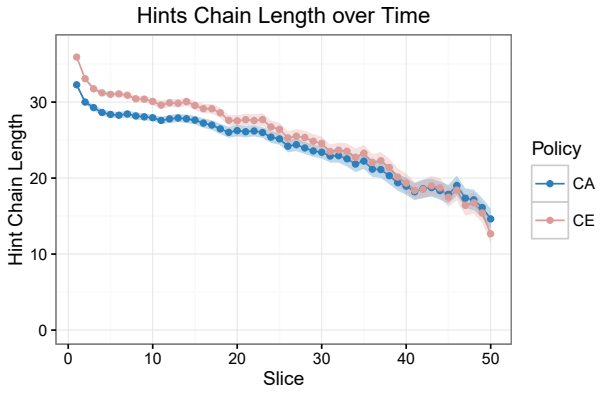


Figure 5: The average CA and CE hint chain length across all snapshot slices. The shading indicates standard error.

objectives. These results suggest that both CTD policies can lead students to high-quality (though sometimes imperfect) final solutions, but exemplar data may be required to generate consistently correct solutions. It is important to note that CTD operates without test cases, and therefore cannot guarantee correctness 100% of the time.

4.2 Objective Satisfaction

To address RQ2, we tested how frequently an available hint would complete an assignment objective before the student did. Figure 6 shows, for each policy in Section 3.1, the percentage of students who had an objective completing hint available for each objective. All hint policies perform fairly well, with at least 45% of students having a completion hint available for objectives 3-9. The CTD policies perform much worse on Objective 2, but otherwise they generally keep pace with the Direct policies. Since these Direct policies offer *all* edits towards their target solution as hints, they should discover most of the possible completing hints. However, it is important to remember that it is not always possible to complete an objective before a student because hints cannot add more than one node to the AST at a time, while a student's edit might change many nodes at once by dragging and dropping code.

It is not sufficient for a hint policy to generate good hints; it is equally important that it *not* generate *bad* hints. To evaluate this second facet of RQ2, we tested how frequently hints from each policy undid an objective, meaning the objective was satisfied before applying the hint, but it became unsatisfied afterward. Figure 7 compares each policy, showing the percentage of students who received a hint that would undo each objective. Predictably, the DS policy, which anticipates a student's final solution, performs well across the board. However, the difference between the DE and CE policies is clear. The CE policy stays below 40% on all objectives, and performs as well or better than the DE policy on all but one objective, often by a factor of 2 or more. The CA policy performs slightly worse than the CE policy on most objectives, most notably Objective 1. This can be attributed to the fact that many students did not in fact complete Objective 1, leading the CA policy to suggest removing the code that did so.

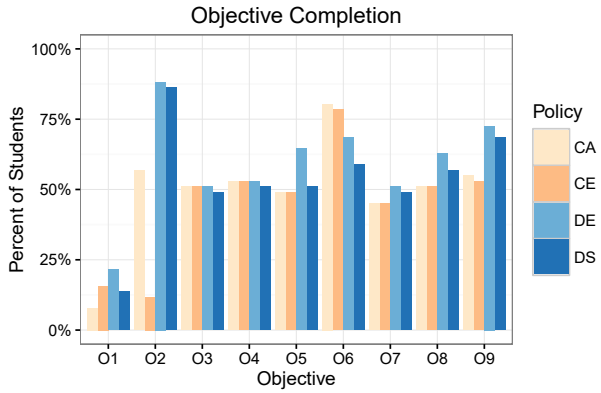


Figure 6: The percent of students who received a hint that completed an objective before the student did under each policy.

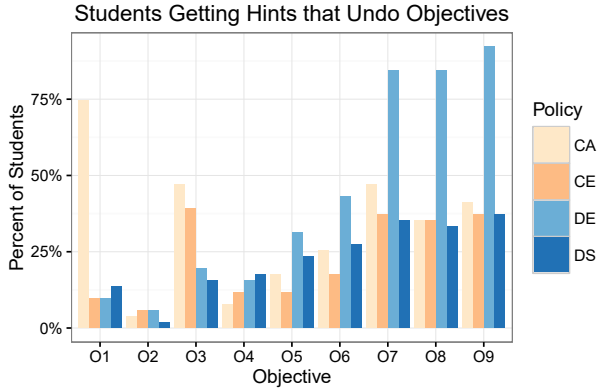


Figure 7: The percent of students who received a hint that undid an objective under each policy.

We do not make the claim that a good hint always completes an assignment objective, nor that undoing an objective always constitutes a bad hint. Still, these criteria serve as good baseline standards for a hint policy. While all policies are fairly successful at suggesting hints that move students toward completing objectives, the CTD and DS policies avoid undoing objectives much better than the DE policy.

4.3 Alignment with Student Actions

RQ3 asks whether or not CTD produces hints which are consistent with a student's solution path. Ideally, a hint policy should not only provide hints which lead to a good solution; as much as possible, these hints should also make sense to the student receiving them. While the comprehensibility of a hint is impossible to measure without user data, we can approximate this by asking whether or not a hint gets the student closer to their future final solution. Presumably, such hints will seem reasonable to the student, as the student eventually went in that direction on their own.

To answer this question, we examined each hint generated with each policy across all code snapshots and calculated whether or not each hint would get the student closer to their final solution than their original state. We used the Robust Tree Edit Distance algorithm [13] to measure the distance between snapshot ASTs. This metric counts the number of insertions, deletions and relabelings required to transform one AST into another. As a baseline, we also calculated this measure for the student's own next state, to determine how frequently a student's actions got them closer to their own final solution state. The results for each policy, averaged over students, are presented in Table 1.

Policy	Closer	SD
CTD All	35.47%	17.50%
CTD Exemplar	32.52%	15.88%
Direct Expert	21.49%	10.02%
Direct Student	39.37%	13.60%
Student Next	60.97%	8.42%

Table 1: The percent of hints under each policy that would bring the student closer to their final solution, averaged over students. Student Next refers to the student's actual next action.

As a baseline, we see that the student's own next step got closer to their final solution 60.95% of the time. The DS policy, which attempts to directly transform the student's state into their solution state, achieves only 39.37%, in part because its hints will often delete useful code and later add it again in a better location. However, the DS policy's performance might be seen as a high target, as it requires future knowledge of the student's actions. In comparison, we see that the CTD policies both approach the DS policy and far outperform the DE policy. The CE policy gets students closer to their final objective 53.4% as frequently as the student's own actions and 82.6% as frequently as the DS policy, and the CA policy performs even better. Post hoc paired t-tests showed that the difference between the CA and DS policies was not significant ($t(50) = -1.63$; $p = 0.109$), while the difference between the CA and DE policies was significant ($t(50) = 6.96$; $p < 0.001$). Interestingly, the difference between the CA and CE policies was also significant ($t(50) = 2.67$; $p = 0.010$), suggesting that restricting data to exemplar students makes CTD hints less reflective of real student behavior. While all policies present some hints that move the student farther away from their final solution, the CTD and DS policies seem to minimize this behavior.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a novel algorithm called CTD for generating next-step hints for students working on open-ended programming assignments. Using data from 51 students working on one such assignment, we have shown that the hints generated by the CTD hint policies can get a student to a high-quality solution from all observed states. We have also shown that the hints are capable of helping students accomplish most assignment objectives before they would otherwise do so, without presenting many hints which undo these objectives. Further, CTD produces hints which get students closer to their final solutions relatively frequently. We have also compared the CA policy, which uses all student data, to the CE policy, which uses exemplar data only. While both policies perform well, the CA policy aligns closer with real student actions, while the CE policy produces higher quality final solutions and is less likely to suggest undoing assignment objectives.

Despite these positive initial results, much work remains to be done to improve CTD. A major limitation of this work is the reliance on a single assignment for evaluation. Future work will explore the efficacy of CTD with a variety of assignments. One challenge that will be presented by larger assignments is ensuring that the contextual goal matching features discussed in Section 2.3 work for programs with multiple scripts. Additionally, while CTD incorporates some of the strategies of the other hint generation algorithms discussed in Section 1.2, such as canonicalization, there are others, such as Rivers and Koedinger's path construction [22], which could also be incorporated. Because the CINs are simply small interaction networks, any advances to the Hint Factory can also be applied to them. Lastly, we have already incorporated our hints into the Snap environment, and future work will investigate how they impact real students. We will explore the effect of CTD hints on students' performance on assignments, as well as their learning gains.

6. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1432156.

7. REFERENCES

[1] T. Barnes and J. Stamper. Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data. In *Proc. of the 9th Int. Conf. on Intelligent Tutoring Systems*, pages 373–382, 2008.

[2] S. Cooper, Y. J. Nam, and L. Si. Initial Results of Using an Intelligent Tutoring System with Alice. In *Proc. of the 17th Annual ACM ITiCSE Conf.*, pages 138–143. ACM Press, 2012.

[3] A. Corbett. Cognitive Computer Tutors: Solving the Two-Sigma Problem. In *Proc. of the 8th Int. Conf. on User Modeling*, pages 137–147, 2001.

[4] A. Corbett and J. Anderson. Locus of Feedback Control in Computer-Based Tutoring: Impact on Learning Rate, Achievement and Attitudes. In *Proc. of the SIGCHI Conference on Human Computer Interaction*, pages 245–252, 2001.

[5] W. Dann, D. Cosgrove, and D. Slater. Mediated transfer: Alice 3 to Java. In *Proc. of the 43rd Annual ACM SIGCSE Conf.*, pages 141–146, 2012.

[6] M. Eagle, M. Johnson, and T. Barnes. Interaction Networks: Generating High Level Hints Based on Network Community Clustering. In *Proc. of the 5th Int. Conf. on Educational Data Mining*, pages 164–167, 2012.

[7] D. Fossati, B. Di Eugenio, S. Ohlsson, C. Brown, and L. Chen. Data Driven Automatic Feedback Generation in the iList Intelligent Tutoring System. *Technology, Instruction, Cognition and Learning*, 10(1):5–26, 2015.

[8] D. Garcia, B. Harvey, and T. Barnes. The Beauty and Joy of Computing. *ACM Inroads*, 6(4):71–79, 2015.

[9] W. Jin, T. Barnes, and J. Stamper. Program Representation for Automatic Hint Generation for a Data-driven Novice Programming Tutor. In *Proc. of the 11th Int. Conf. on Intelligent Tutoring Systems*, pages 1–6, 2012.

[10] T. Lazar and I. Bratko. Data-Driven Program Synthesis for Hint Generation in Programming Tutors. In *Proc. of the 12th Int. Conf. on Intelligent Tutoring Systems*, pages 306–311. Springer, 2014.

[11] B. Moskal, D. Lurie, and S. Cooper. Evaluating the Effectiveness of a New Instructional Approach. *ACM SIGCSE Bulletin*, 36(1):75–79, 2004.

[12] T. Murray. Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art. *Int. Journal of Artificial Intelligence in Education*, 10:98–129, 1999.

[13] M. Pawlik and N. Augsten. RTED: A Robust Algorithm for the Tree Edit Distance. *Proceedings of the VLDB Endowment*, 5(4):334–345, 2011.

[14] B. Peddycord III, A. Hicks, and T. Barnes. Generating Hints for Programming Problems Using Intermediate Output. In *Proc. of the 7th Int. Conf. on Educational Data Mining*, pages 92–98, 2014.

[15] D. Perelman, S. Gulwani, and D. Grossman. Test-Driven Synthesis for Automated Feedback for Introductory Computer Science Assignments. In *Proc. of the Workshop on Data Mining for Educational Assessment and Feedback*, 2014.

[16] C. Piech, M. Sahami, J. Huang, and L. Guibas. Autonomously Generating Hints by Inferring Problem Solving Policies. In *Proc. of the 2nd ACM Conf. on Learning @ Scale*, pages 1–10, 2015.

[17] T. W. Price and T. Barnes. An Exploration of Data-Driven Hint Generation in an Open-Ended Programming Problem. In *Proc. of the Workshop on Graph-Based Data Mining held at EDM'15*, 2015.

[18] T. W. Price and T. Barnes. Comparing Textual and Block Interfaces in a Novice Programming Environment. In *Proc. of the Int. Computing Education Research Conference*, 2015.

[19] M. Resnick, J. Maloney, H. Andrés, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai. Scratch: Programming for All. *Communications of the ACM*, 52(11):60–67, 2009.

[20] K. Rivers and K. Koedinger. A Canonicalizing Model for Building Programming Tutors. In *Proc. of the 11th Int. Conf. on Intelligent Tutoring Systems*, pages 591–593, 2012.

[21] K. Rivers and K. Koedinger. Automatic Generation of Programming Feedback: A Data-driven Approach. In *Proc. of the First Workshop on AI-supported Education for Computer Science*, pages 50–59, 2013.

[22] K. Rivers and K. R. Koedinger. Data-Driven Hint Generation in Vast Solution Spaces: a Self-Improving Python Programming Tutor. *Int. Journal of Artificial Intelligence in Education*, 2015.

[23] R. Singh, S. Gulwani, and A. Solar-Lezama. Automated Feedback Generation for Introductory Programming Assignments. *ACM SIGPLAN Notices*, 48(6), 2013.

[24] J. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental Evaluation of Automatic Hint Generation for a Logic Tutor. *Int. Journal of Artificial Intelligence in Education*, 22(1):3–17, 2013.

[25] I. Utting, S. Cooper, and M. Kölling. Alice, Greenfoot, and Scratch – A Discussion. *ACM Transactions on Computing Education*, 10(4), 2010.

[26] K. Zimmerman and C. R. Rupakheti. An Automated Framework for Recommending Program Elements to Novices. In *Proc. of the 30th Int. Conf. on Automated Software Engineering*, 2015.

How to Model Implicit Knowledge? Similarity Learning Methods to Assess Perceptions of Visual Representations

Martina A. Rau
Educational Psychology
University of Wisconsin—Madison
1025 W. Johnson St
Madison, WI 53706
marau@wisc.edu

Blake Mason
Electrical and Computer Engineering
University of Wisconsin—Madison
1415 Engineering Dr
Madison, WI 53706
bmason3@wisc.edu

Robert Nowak
Electrical and Computer Engineering
University of Wisconsin—Madison
1415 Engineering Dr
Madison, WI 53706
nowak@ece.wisc.edu

ABSTRACT

To succeed in STEM, students need to learn to use visual representations. Most prior research has focused on conceptual knowledge about visual representations that is acquired via verbally mediated forms of learning. However, students also need perceptual fluency: the ability to rapidly and effortlessly translate among representations. Perceptual fluency is acquired via non-verbal, implicit learning processes. A challenge for instructional interventions that focus on implicit learning is to model students' knowledge acquisition. Because implicit learning is non-verbal, we cannot rely on traditional methods, such as expert interviews or student think-alouds. This paper uses similarity learning, a machine learning method that can assess how people perceive similarity between visual representations. We used this approach to model how undergraduate students perceive similarity between visual representations of chemical molecules. The approach achieved good accuracy in predicting students' similarity judgments and expands expert predictions of how students might perceive visual representations of molecules.

Keywords

Perceptual knowledge, implicit learning, visual representations, similarity learning methods, chemistry.

1. INTRODUCTION

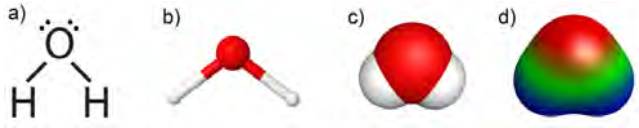


Figure 1. Visual representations of chemical molecules: a: Lewis structure; b: ball-and-stick model; c: space-filling model; d: electrostatic potential map (EPM) of water.

Visual representations are ubiquitous instructional tools in science, technology, engineering, and math (STEM) domains [1, 2]. For example, instructors use the visual representations shown in Figure 1 to help students learn about chemical bonding. Yet, to a novice student, these visual representations may not be helpful because the student may not know how to interpret the representations. For instance, does the red color in the ball-and-stick figure (Figure 1-b) mean the same thing as in the electrostatic potential map (EPM; Figure 1-d)? (It does not.)

Instructors often ask students to use visual representations that they have never seen before to make sense of concepts that they have not yet learned about [3, 4], an issue known as the *representation dilemma* [5]. Hence, to succeed in STEM, students need *representational competencies* that enable them to use visual representations to make sense of and solve domain-relevant problems [6, 7]. One crucial representational competency is the ability to interpret visual representations; that is, to map visual representations to the abstract concepts they depict [6, 8]. For example, students need to understand how the representations in Figure 1 show information about the molecule. For the Lewis structure (Figure 1-a), the student may map the unbonded electrons shown as dots to conceptual knowledge about how polarity in chemical molecules and infer that the water molecule has a local negative charge by the Oxygen atom.

Educational technologies are particularly suitable to support representational competencies because they can provide adaptive support while students solve domain-relevant problems [9, 10]. Such adaptive support relies on a cognitive model that infers whether the student has learned target skills based on her/his interactions with the technology. Research shows that adapting instruction to students' representational competencies can enhance those competencies [11] and learning of domain knowledge [12].

However, educational technologies for representational competencies have two critical limitations. First, they typically focus on one set of representational competencies: students' conceptual understanding of representations (e.g., the ability to explain how visual features depict concepts). This focus mimics education psychology research's focus on conceptual learning [6, 13]. Conceptual knowledge is invariably intertwined with a second type of representational competency: *perceptual knowledge* [14, 15]; the ability to rapidly and effortlessly recognize conceptual information based on visual features of the representations. This ability results from *implicit forms of learning*. For example, expert chemists simply "see" that the molecules depicted in Figure 1 have a local negative charge by the Oxygen atom, without having to make a an effortful conceptual inference.

Second, of the few educational technologies that enhance perceptual fluency, their adaptive capabilities are limited and their perceptual supports rely solely on performance measures (e.g., accuracy, response times) to adapt to students' representational competencies [15, 16]. They do not use a cognitive model of the latent skills that students acquire through perceptual learning. As a result, they cannot provide specific feedback when students make mistakes. Decades of research showing that cognitive models can dramatically increase the effectiveness of educational technologies [10, 17] suggest that we must address this limitation and create adaptive instruction for perceptual knowledge.

These limitations likely result from cognitive modeling's traditional focus on explicit, verbally accessible knowledge. To develop cognitive models, researchers analyze how students think about target skills [9, 18]. We typically ask students to verbalize their problem-solving steps [19, 20]. Yet, verbalization is not suitable for assessing perceptual learning processes, which are implicit and not verbally accessible [14, 21]. Therefore, instructional designers have to rely on "educated guesses" as to which visual features students may pay attention to. These educated

guesses are based on the novice-expert literature, which documents the fact that novices tend to rely on surface features; that is, easily perceivable visual cues such as color and shape, to judge the similarity between stimuli items. By contrast, experts rely on visual features that are conceptually relevant and hence make more refined distinctions between visual features. Thus, to create adaptive perceptual supports, we need to develop cognitive models for perceptual learning.

Our research takes a first step towards developing a cognitive model for perceptual learning by assessing students' perceptual knowledge of a common visual representation in chemistry. In particular, we investigate *research question 1:* Which visual features do students focus on when presented with visual representations? To address this question, we asked hundreds of students to judge the similarity between visual representations of molecules. We then used *similarity learning*—a machine learning method that provides a formal approach to investigating how people perceive similarity among visual stimuli. This method allowed us to estimate latent factors that account for the perceived similarity relationships between representations. Because we can map these latent factors to the visual features in the representations, this approach allows us to investigate which visual features are most salient to students' perceptions of similarity. Comparing these visual features to "educated guesses" allowed us to test *research question 2:* Do the visual features we identified as salient via metric learning correspond to visual features that students are expected to attend to based on the expert-novice literature on perceptual learning? In addition, we investigated a methodological *research question 3:* How many similarity judgments we need to assess students' perceptual knowledge?

Although we address these questions in the context of a particular domain with a particular visual representation, this paper makes two important broader contributions. First, it provides an empirical validation of the "educated guesses" that developers of perceptual learning technologies typically rely on. Second, it establishes a methodology to assess perceptual knowledge that can serve as a basis for a cognitive model of perceptual learning. These contributions build the foundation for the development of adaptive instruction for perceptual knowledge and other implicit knowledge.

2. EXPERIMENT

2.1 Visual Representations of Molecules

For our experiment, we selected visual representations of chemical molecules common in undergraduate instruction. Lewis structure representations are the most commonly used visual representations in undergraduate chemistry textbooks. We reviewed textbooks and online instructional materials and listed the frequency of all occurring molecules using their chemical names (e.g., H_2O) and common names (e.g., water). For our experiment, we chose the 50 most common molecules.

First, we created *educated guess features* (Figure 2, yellow) that correspond to expert assessments of which visual features students may attend to when making similarity judgments. To obtain these educated guesses, we reviewed the literature on chemistry expertise [22, 23] and on perceptual learning [14, 24], and conducted learner-centered interviews with undergraduate and PhD students in chemistry [25]. We identified 6 educated guess features: number of total letters, number of distinct letters, number of total bonds, number of single bonds, number of unbonded electrons, and molecule geometry (linear, planar, tetrahedral).

To investigate which visual features drive students' similarity judgments, we quantitatively described the visual features of the

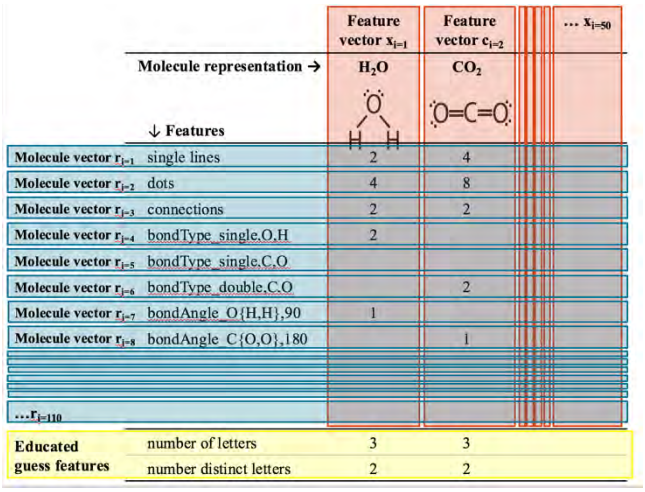


Figure 2. Example features for H_2O and CO_2 molecule representations with educated guess features in yellow, feature vectors in red, and molecule vectors in blue.

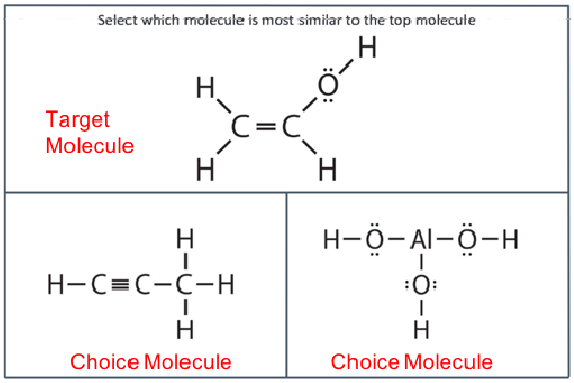


Figure 3. Example of a similarity judgment task: given the molecule on the top, students were asked which of the two molecules at the bottom is most similar.

molecule representations. To this end, we created *feature vectors* for each of the molecules (see Figure 2, red) that describe which visual features the representation contains (e.g., bond angles, the numbers of specific atoms, or the numbers of different atoms present). The feature vectors of our corpus of molecule representations contained a total of 110 features. The 50 feature vectors collectively form matrix $X = [x_1, x_2, x_3, \dots, x_{50}]$, where x_i, is the feature vector for the *i*th molecule.

We aggregated each element of the feature vectors into *molecule vector* for individual features (Figure 2, blue). Each molecule vector consisted of 50 values describing how many times the feature occurred in each representation. As molecule vectors make up the rows of our matrix of 110 features by 50 molecules shown in Figure 2, we will refer to the molecule vector for the *j*th feature as r_j. Thus, feature vectors provide a numeric description of the visual information present in each representation, whereas molecule vectors provide a numeric description of overall patterns of visual features in the dataset for all representations.

2.2 Similarity Judgment Tasks

Students completed similarity judgment tasks that were presented as triplet comparisons (see Figure 3). Given a representation of a molecule (the "target-molecule"), students were asked to choose molecules") was most similar to the given one. For each task, the student chose between one of the two choice-molecules that

he/she perceived to be more similar to the target-molecule. After each task, another triplet was generated uniformly at random from our corpus of molecule representations.

We delivered the similarity judgment tasks via NEXT; a cloud-based machine learning platform [26]. NEXT allows users to upload their own content and query participants to perform judgment tasks. It uses machine learning algorithms to automate data collection and analyze results. More information about the platform can be found at http://nextml.org. In NEXT, students first received a brief description of the study and then worked through a sequence of 50 similarity judgment tasks. Students were instructed that these tasks are not a test and that there is right or wrong answer, but that we they are simply asked about their personal perceptions of similarities among molecule representations.

2.3 Dataset

Undergraduate students enrolled in an introductory chemistry course at a large U.S. university were invited to participate in a survey on learning with visual representations. The course had an enrolment of 781 students. Participation was voluntary. Altogether, we collected 26,180 responses from 563 (possibly non-unique) students. 61.6% of the students completed all 50 similarity judgment tasks. On average, students completed 46.5 tasks. Each similarity judgment in response to a triplet comparison task was associated with the feature vectors (x_i) and molecule vectors (r_j) of the three molecule representations, as described in 2.1.

3. ANALYSIS

In the following, we describe how we used similarity learning to investigate which visual features drive students' similarity judgments. We first provide a brief introduction into the metric learning method in general. Then, we describe how we applied this method to our dataset in particular.

3.1 Introduction to Similarity Learning

In general, the goal of similarity learning is to learn a similarity function *f* that agrees with students' similarity judgments in the following sense: if item *i* is judged to be more similar to *j* than to *k*, then f(i,j) < f(i,k). The function *f* can be thought as quantifying the perceived distance or dissimilarity between pairs. Alternatively, the function could quantify the perceptual similarity (inverse distance) between pairs, in which case f(i,j) > f(i,k).

People are better at providing ordinal (i.e., comparative) responses than at providing fine-grained quantitative judgments or ratings [27]. For example, when asked to compare the visual representations in Figure 3, people find it easier to judge whether the target molecule is more similar to the left or the right choice molecule than to judge their similarity on a rating scale. However, it is challenging to machine-learn embeddings from comparisons due to the sheer number of possible triplet comparisons that could be made; the number of distinct triplets is proportional to n^3. For example, in our case of n=50 molecule representations, there exist nearly 125,000 distinct triplets. Researchers have observed that while triplet comparisons are easy to answer, they can become tedious and boring after extended sessions [28]. Since we hypothesize that perceived dissimilarities can be accurately represented in d-dimensional space, it is reasonable to conjecture that if the embedding dimension is low (i.e., $d \ll n$), then there will be a high degree of redundancy among the triplet comparisons. In fact, researchers have observed that a small subset of these triplet comparisons often suffice to learn a reasonably accurate embedding, lending support to this conjecture [29-31].

3.2 Similarity Learning Approaches

We applied two similarity learning approaches in this paper: similarity learning by ranking [32] and non-metric multi-dimensional scaling. In both cases, we modelled the perceptual similarity between molecules *i* and *j* as

$$S_{ij} = \mathbf{x}_i^T \mathbf{A} \mathbf{x}_j$$

Here $\boldsymbol{A}$ is a symmetric matrix that parameterizes the model. The k,lth element of the matrix, denoted by A_{kl} represents the importance of the interaction of feature k and feature l in the model. Since we assume $\boldsymbol{A}$ is symmetric, $A_{kl} = A_{lk}$ and $S_{ij} = S_{ji}$.Before introducing these approaches, let us define some notation. There are N triplet comparisons. For the nth triplet, let i_n denote the target-molecule and let j_n and k_n denote the two choice-molecules. Let y_n denote the student's judgment, specifically $y_n = +1$ if the student decided j_n was more similar to i_n and $y_n = -1$ otherwise. Each of the $p = 50$ diagrams also has m associated features (e.g., numbers of different atoms, bonds, etc.). Arrange the features for each molecule representation into an m x 1 molecular feature vector, and the m x 1 feature vectors into a m x p matrix, X. The ith column of X, denoted x_i, contains the m features for molecule i. The jth row of X, denoted r_j, is a molecule vector for feature j containing the value of feature j for all 50 representations.

3.2.1 Approach 1: Similarity Learning by Ranking

This approach learns matrix A in our model of perceptual similarity directly from triplet responses via linear regression.

$$S_{ij} = \mathbf{x}_i^T \mathbf{A} \mathbf{x}_j$$

where x_i and x_j are m x 1 dimensional feature vectors of the m features of molecule representations i and j. The matrix A is m x m, and the metric learning problem is to estimate A that minimizes the number of disagreements between the ranking predictions for each triple (i.e., either $S_{ij} > S_{ik}$ or vice-versa) and the comparative judgments collected from the students, as proposed by [32].

The first step in this analysis was to estimate A. Formally, the estimation of A can be written as the following optimization problem. Let $\mathbb{S}_m$ be the set of all m x m symmetric matrices. Solve for A that minimizes:

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A} \in \mathbb{S}_m} \sum_{n=1}^{N} \left(y_n - \mathbf{x}_{i_n}^T \mathbf{A} \mathbf{x}_{j_n} + \mathbf{x}_{i_n}^T \mathbf{A} \mathbf{x}_{k_n} \right)^2$$

where the superscript T denotes the vector transposition. The matrix A that minimizes the sum of squared errors weights the similarities between the diagram features so as to predict perceptual similarity judgments. In general, the solution A will place some weight on all m features. We anticipate that the visual features that are not salient do not strongly affect students' similarity judgments and therefore have lower weights in A.

Taking this thinking a step further, we could consider many different optimizations of the type above, where in each case we use different subsets of the features, in order to determine which are most predictive of student judgments. Indeed, some features maybe totally irrelevant and worsen, rather than help, the prediction of students' similarity judgments. Unfortunately, searching over all possible subsets of features is computationally infeasible, so we instead consider the following optimization that approximates this search problem called sparse COMET [33].

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}\in\mathbb{S}_m} \sum_{n=1}^{N} \left(y_n - \mathbf{x}_{i_n}^T \mathbf{A} \mathbf{x}_{j_n} + \mathbf{x}_{i_n}^T \mathbf{A} \mathbf{x}_{k_n}\right)^2 + \lambda \sum_{k=1}^{m} ||\mathbf{A}(k,:)||_2^2$$

This optimization method uses a cost function that consists of two terms. The first term represents least squares data-fitting cost in the previous optimization. The second term is a Group LASSO penalty, which encourages solutions that have many columns equal to 0. If a column in *A* is all zero, then the corresponding feature is not used for prediction. The number of zero-valued columns in the solution depends on $\lambda > 0$. Note that we recover the previous optimization when $\lambda = 0$. Larger values of λ produce sparser solutions that effectively use fewer features. Features crucial for prediction are excluded only if λ is exceedingly large.

The second step in this analysis was to tune the parameter λ and then to assess the prediction accuracy of our method. To this end, we used 10-fold cross validation. Specifically, we randomly split the complete dataset into 10 equal sized subsets. We removed 2 random subsets as hold-out data and kept the remaining data as training data. We then solved the optimization above with the training data over a range of different λ values. For each λ, we scored prediction accuracy on one set of hold-out data to select the optimal value. Then, using our chosen λ value, we solved the optimization again to obtain a final *A* using 9/10 of the data, and assessed the prediction accuracy on remaining 1/10 of the data.

The final step was to rank the features based on the weights in matrix. Due to the Group LASSO penalty in the loss function, many of the columns in the resulting matrix are zero. To get the aggregate weight of each relevant feature, we computed the length (norm) of each non-zero column and ranked accordingly.

3.2.2 Approach 2: Ordinal Embedding

In this approach, rather than directly making predictions of similarity based on feature vectors and triplet responses, we first used students' similarity judgments to learn an embedding that spatially represents the similarity of molecule representations as distances in 2-dimensional space. We then identified molecule vectors that account for the distribution of molecule representations in the embedding space.

The first step in this analysis was to learn an embedding. We applied non-metric multidimensional scaling (NMDS) to the 26,180 triplet comparison responses collected from the experiment to learn an embedding of the 50 molecule representations in a two-dimensional space [22]. Embedding in two dimensions allows visualizing the perceived similarity computed by NMDS. The embedding reflects the consensus among students as to which molecular representations were more or less similar. We created 50 different embeddings, using multiple random initializations per embedding in order to account for the non-convexity of NMDS.

The second step was to validate the embedding. To this end, we computed a distance matrix for each embedding. To validate the distance matrices, we used the following cross-validation procedure. We selected 6000 triplet comparison responses uniformly at random to serve as a hold-out dataset. From the remaining triplets, we randomly selected training sets of different size, ranging from 1000 to 20,000 triplet comparison responses. We computed embeddings for each training set. We then used these embeddings and the associated distance matrices to predict students' similarity judgments. Next, we used the distances in the embedding as a predictor of judgments in the hold-out set; the prediction errors quantify how well the embedding reflects the judgments. We repeated this procedure for training sets of different size. We performed 50-fold cross validation to calculate average prediction error on the learned embeddings. This procedure allowed assessing how prediction performance relates to the training set size (i.e., how many triplets were used to compute an embedding).

The third step in our analysis, after validating our embedding procedure, was to compute an embedding and corresponding distance matrix from the full set of triplets. Since the distance between points in the embedding corresponds to their perceived dissimilarity, we computed a similarity matrix defined as the element-wise inverse of the distance matrix, scaled from 0 to 1.

The fourth step was to identify which features, represented by the feature vectors, drive students' similarity judgments. Because the embedding was performed in 2 dimensions, we can consider the problem of only choosing 2 feature vectors to combine and compare combinations of pairs of feature vectors to the similarity matrix. For each possible pair, we performed a least squares optimization to find the ideal uniform scaling to match an outer product of our feature vectors to the similarity matrix.

$$\widehat{A} = \arg\min_{\mathbf{A}} \sum_{i,j=1}^{p} \left(S_{ij} - \mathbf{x}_i^T \mathbf{A} \mathbf{x}_j\right)^2$$

subject to $A_{st} = 0$ for all *s,t* not equal to *k,l* or *l,k*. In other words, only let the *k,l* elements of *A* be non-zero and optimize these. This equates to fitting *S* to the molecule vectors for features *k* and *i*. Here, S_{ij} represents the value of the perceptual similarity between molecules *i* and *j* from the embedding. The magnitude of resulting value of A_{kl} tells us how important the interaction of features *k* and *l* is in representing the similarity. This is basically a correlation coefficient, and it only gauges the marginal value of this interaction (i.e., in isolation of all other interactions). In each case, after learning a matrix *A_* we computed the corresponding residual value between similarity matrix *S* and our combination of 2 features. After performing all possible combinations of pairs of features, we ranked pairs of features in ascending order of residual values, with the smallest residuals being the best approximation of our observed similarity matrix. To evaluate the feature rankings, we used 10-fold cross-validation by performing identical tests on 10 different similarity matrices computed from different embeddings based on equal numbers of triplets to ensure that the original embedding and the non-convexity of NMDS was not a factor in the final ranking of feature pairs.

4. RESULTS

4.1 Identifying Important Visual Features

To address *research question 1*, we used the two similarity learning approaches just described to identify which visual features account for students' similarity judgments.

4.1.1 Approach 1: Similarity Learning by Ranking

Recall that the first approach entailed learning a similarity function that describes students' perceived similarity between molecule representations. This approach yielded an average 69% prediction accuracy of students' similarity judgments (assessed via 10-fold cross validation). This finding indicates that there was consensus over which representations were more or less similar, but also that there were some disagreements among students' similarity judgments.

To identify which visual features account for students' similarity judgments, we estimated the weights for each feature in the ma-

Table 1. Top 10 features from the ranking of features with strong weights obtained by Approach 1.

Feature	Avg weight
Distinct letters	4.50%
Single bonds between Oxygen and Hydrogen	3.45%
180-degree angle in Hydrogen-Carbon-Fluorine	3.16%
Double bonds between Oxygen and Nitrogen	3.03%
Number of Nitrogen atoms	2.99%
Double bonds between Carbon and Oxygen	2.78%
120-degree angle in Hydrogen-Carbon-Hydrogen	2.73%
Number of Oxygen atoms	2.64%
180-degree angle in Carbon-Carbon-Oxygen	2.62%
Single bonds between Carbon and Oxygen	2.37%

chine-learned matrix *A*. The stronger a feature's weight in *A*, the more this feature affected students' similarity judgments. Hence, the feature's weight corresponds to its saliency in students' perception of molecule representations.

Table 1 shows the 10 most important features, as determined by a ranking of features according to their aggregate weight computed from matrix *A*. These results show that the most highly ranked feature is the number of distinct letters, which corresponds to an aggregate educated guess feature. Specific visual features that are relevant to organic molecules were also ranked highly (e.g., the number of single bonds between Oxygen and Hydrogen atoms, the number of bonds between Carbon and Oxygen, the number of Nitrogen and Oxygen atoms). These specific visual features were present in many of the molecules in our dataset. Several visual features also included geometric aspects, specifically bond angles. These features indicate the presence of chemical functional groups that are relevant to predicting molecule's reactive behaviors.

4.1.2 Approach 2: Ordinal Embedding

Recall that approach s learns an embedding that represents the similarity of molecule representations as distances in a d-dimensional space, from which we then extracted the most important features. First, we established how many dimensions we need to consider (i.e., which d to choose in representing similarity of molecule representations in a d-dimensional space). Using the process of 50-fold cross validation described above, we calculated unit through 20 dimensional embeddings of perceptual similarity. We used 20,000 triplets in this computation to ensure that the number of triplets did not affect the prediction accuracy as the dimension became large. Figure 4 shows that there is no drop in prediction accuracy when embedding in low dimensions versus high, suggesting that perceptual similarity can be accurately represented in a low dimensional subspace, and that there is a high degree of redundancy in the data. This result shows that students' responses agreed on the relative similarity about 70% of the time.

Next, we generated a 2-dimensional embedding that describes students' perceived similarity between the molecule representations. Figure 5 shows this embedding, illustrating that molecules naturally form clusters based on their perceptual similarity. These clusters correspond to specific chemical properties shared among the molecules, such the presence of a particular type of bond or a functional group. We color-coded and labeled some of these clusters to illustrate these characteristics of students' perceptions. This illustration lends face validity to our embedding approach.

From this embedding, we extracted an ordered list of the feature pairs that best capture students' similarity judgments, shown in Table 2. The feature pairs in this table were ranked based on how well they approximate the similarity matrix computed from the

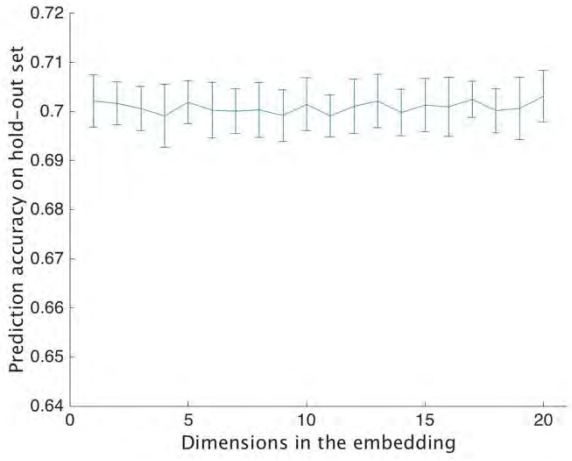


Figure 4. Prediction accuracy on hold-out set by number of dimensions in embedding.

Table 2. Top 10 feature pairs from Approach 2. Each row corresponds to a pair of feature vectors ranked in accordance with how accurately they described the observed similarity structure from the embedding.

Rank	Feature pairs		
1	Distinct letters	&	Distinct letters
2	Total letters	&	Distinct letters
3	Distinct letters	&	Single bonds
4	Total bonds	&	Distinct letters
5	Distinct letters	&	Carbons
6	Hydrogens	&	Distinct letters
7	Total letters	&	Total letters
8	Total letters	&	Single bonds
9	Total letters	&	Unbonded electrons
10	Distinct letters	&	Single Carbon-Hydrogen bonds

embedding in Figure 5. The same feature may appear twice in a pair to account for the possibility that a weighted combination of a feature with itself better reflects the observed similarity structure than does a pair of features. In sum, these results show that the most highly ranked features are general visual features, which correspond to the aggregate educated guess features (e.g., number of letters, number of lines). Specific visual features that are relevant to hydrocarbon molecules were also ranked highly (e.g., the number of Carbon and Hydrogen atoms). These specific features were present in many of the molecules in our dataset.

4.1.3 Comparing the Similarity Learning Approaches

While both methods agreed upon the top ranked feature, the similarity learning by ranking approach ranked structural features of the representations that were relevant to hydrocarbons and organic molecules more highly. As the ranking from this method follow predictive power, this ranking indicates that students' judgments of similarity can best be predicted, and therefore explained, through a combination of the number of different letters and the structural features involving Carbon, Hydrogen, and Oxygen.

4.2 Comparison with "Educated Guesses"

To address *research question 2* (do the visual features we identified as salient via metric learning correspond to visual features that students are expected to attend to?), we compared the results from the similarity learning approaches to the educated guess features that we had determined based on the expert-novice litera-

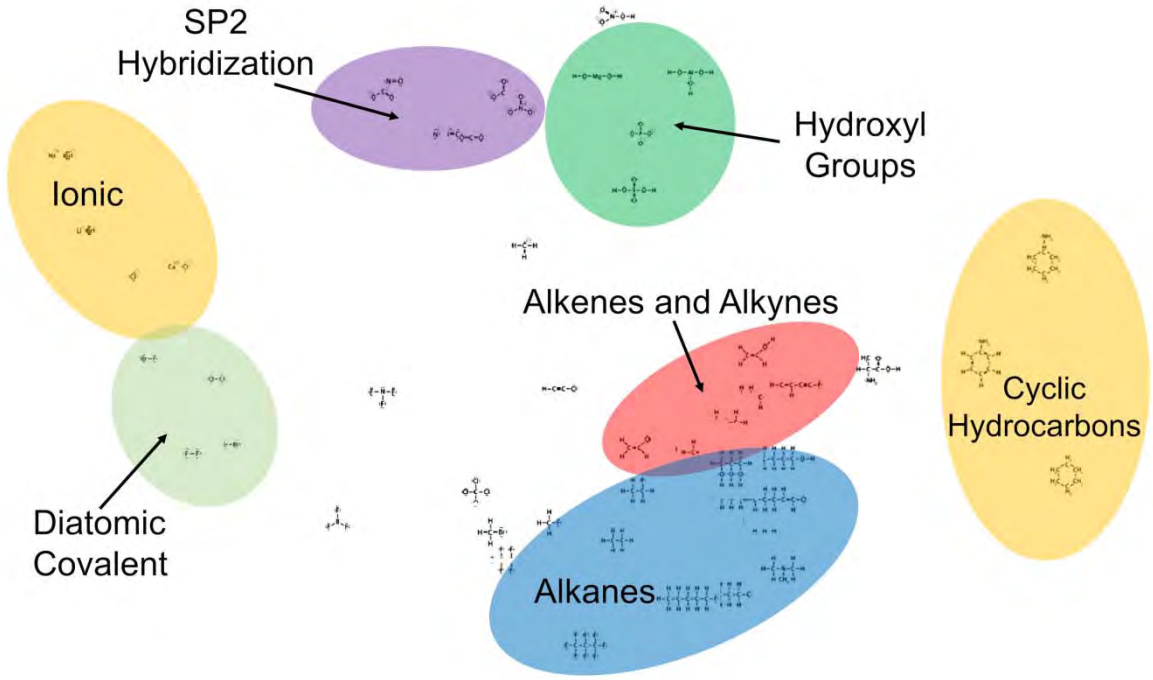


Figure 5. 2-dimensional similarity embedding from Approach 2. Distances between molecule representations correspond to students' perceptions of dissimilarity (i.e., molecule representations that are depicted close to one another are perceived to be similar).

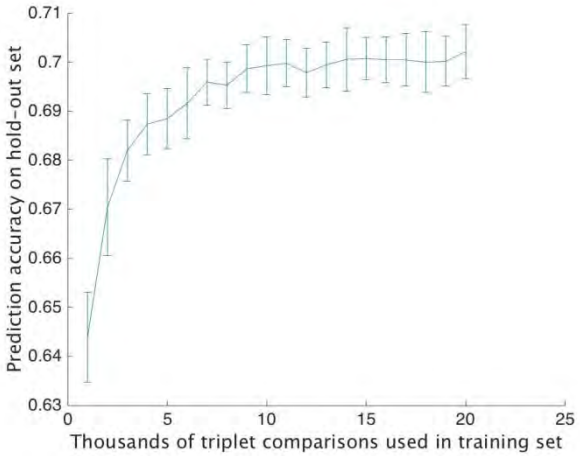


Figure 6. Prediction accuracy on hold-out set by number of triplet comparison judgments used in the training set.

ture on perceptual learning. Overall, the results from both metric learning approaches agree with the educated guesses: aggregate features that describe general visual features were ranked to be most important by both metric learning approaches. The similarity learning by ranking approach also yielded a number of visual features that are specific to the types of molecules in our corpus; in particular, visual representations that are highly relevant for comparing organic molecules.

4.3 Number of Similarity Judgments Needed

We addressed our methodological *research question 3* (how many similarity judgments we need to assess students' perceptual knowledge) with the ordinal embedding approach. Specifically, we tested how many triplet comparisons are required to compute a representative embedding of the underlying similarity. Figure 6 shows that gains in prediction accuracy of the embedding were no longer statistically significant beyond 7000 triplet comparisons.

4.4 Differences Between the Two Approaches

The two methods are different and potentially complementary. There is no definitively correct way to fit the common model $S_{ij} = \boldsymbol{x_i^T A x_j}$ to data. The main differences in the final rankings they produce stems from how we are learning matrix A and the restrictions we put on its structure. In approach 1 we are directly working with triplet responses which are perhaps noisy due to disagreements in students' individual judgments of perceptual similarity, but we are placing fewer restrictions on the learned matrix, allowing for more feature interaction. In approach 2, NMDS is useful for capturing perceived similarity in aggregate, but we enforce much stronger restrictions on the structure of A, namely that only two features may interact at once, giving a clearer picture of the importance of a pair of features.

If we had to recommend one approach, we prefer the regression approach (approach 1) because it optimizes prediction error, which is an objective measure of model quality. The embedding approach (approach 2) has its own potential virtues: The low-dimensional embedding provides an implicit form of regularization that may be helpful especially if the amount of response data is small. Also, the embedding provides a visual representation of perceptual similarities which is helpful for model interpretation.

5. DISCUSSION

We applied similarity learning approaches to assess which visual features students focus on when presented with visual representations. We compared two approaches, one that allows us to assess the predictive power of the identified features, and one that allows representing the perceived similarity in a d-dimensional space. Both approaches yield similar results as to which visual features

are salient to students. Hence, both approaches address research question 1: Which visual features do students focus on when presented with visual representations? We found that students' similarity judgments of Lewis structures appear to be driven by general visual features such as the number of total and distinct letters, as well as by visual features specific to the types of molecules in our dataset (e.g., number of Hydrogen / Carbon atoms).

Our results also address research question 2: Do the visual features we identified as salient via similarity learning correspond to visual features that students are expected to attend to based on the expert-novice literature on perceptual learning? We found that the identified general visual features align with educated guesses based on the literatures on expertise and perceptual learning, which validates the common "educated guess" approach that instructional designers have to rely on in the absence of assessments of perceptual knowledge. Our results also suggest that, in addition to these general features, students learn to pay attention to key visual features that are highly domain-specific; such as features that indicate the presence of functional groups that are predictive of chemical behaviors. Furthermore, our results show that a few key features predict students' perceptions of similarity between visual representations with accuracy of about 70%.

Finally, we addressed our methodical research question 3: How many similarity judgments we need to assess students' perceptual knowledge? Our results show that about 7,000 responses to triplet comparison tasks are sufficient in assessing a population's perceptual knowledge. Using a survey with 50 triplet comparison tasks (as in our experiment), that means an N of 140 participants will yield valid assessments of perceptual knowledge.

6. LIMITATIONS

Although both similarity learning approaches had rigorous theoretical backing, we made a few assumptions about our triplet comparison data that had inherent limitations of note. In both of these methods, we are not modelling individual students, but rather the population as a whole. Consequently, we assume that the triplets and therefore the judgments of similarity are independent of one another. This assumption allows us to learn the rankings of features and feature pairs for the students' collectively, but it does not provide a ranking for an individual. Further, because judging similarity representations is a subjective task, students' judgments may in certain cases conflict with one another. Even with an extremely large number of similarity judgments, complete consensus is unlikely, and therefore, perfect prediction of student judgments is similarly difficult to achieve. Hence, future research needs to investigate how to expand the present approach to modeling individual perceptual knowledge.

Another limitation pertains to the ordinal embedding procedure. For visualization purposes, we embedded the molecules into a 2-dimensional space. Higher dimensional embedding may more accurately capture perceptual dissimilarities. Future research should explore this question.

7. FUTURE DIRECTIONS

We will expand our research to other types of visual representations typically used in chemistry instruction (see Figure 1). Further, we will gather data from expert chemists and compare them to data from novices and advanced learners. Based on this comparison, we will identify a "perceptual knowledge gap" between students and experts. Specifically, we will identify visual features that experts attend to but students do not.

Further, we will expand similarity learning so that it can assess an individual student's perceptual knowledge in real time. The current approach is limited in that it requires a large number of similarity judgments to assess students' perceptual knowledge, which is only feasible if we are interested in assessing perceptual knowledge of a population of interest (e.g., novices, advanced students, experts), and because we assume independence among similarity judgments. To address this limitation, we will combine our similarity learning approach with cognitive modeling methods (e.g., Bayesian knowledge tracing). For example, a similarity judgment survey may provide a prior for in a cognitive model, and students' performance on perceptual learning tasks may inform the choice of representations for a small number similarity judgment tasks interspersed in the learning activity.

This expansion will provide the basis for the design of adaptive instruction for perceptual knowledge that can provide appropriate sequences of perceptual learning tasks that draw students' attention to visual features they yet have to learn. Further, knowing which visual features students have not yet learned can serve as a basis for the design of visual feedback that highlights visual features when students make mistakes on perceptual learning tasks.

In sum, we will use the similarity learning approach described in this paper both to design instruction for perceptual learning and to assess perceptual knowledge as a learning outcome.

8. CONCLUSIONS

This paper described a new approach to assess students' perceptual knowledge. We used this approach to validate the "educated guesses" approach. In addition, we offer more formal pathways for instructional designers to create perceptual learning assessments. Because developing adaptive instruction for perceptual knowledge relies on such assessments, this paper makes an important contribution to cognitive modeling research.

This paper also makes important contributions to machine learning. We provide a new mathematical approach to quantify the accuracy of perceptual embeddings learned from similarity judgments. Specifically, we derived bounds on the accuracy of embeddings learned from small numbers of comparative judgments by adapting recently developed large-sample analysis methods [34]. This approach provided new algorithms for generating embeddings that are provably accurate. We investigated new methods for embedding based on spectral methods inspired by spectral ranking algorithms [35]. Our experiment yielded an empirical validation with perceptual data from undergraduates, as well as new machine learning methods to assess how visual features predict or encode perceptual similarity judgments. Specifically, we explored the application of group Lasso algorithms for automatically selecting the most perceptually salient features [36]. Our experiment empirically evaluated the group Lasso approach.

In sum, our work provides a crucial stepping stone towards adaptive instruction for perceptual knowledge. Perceptual knowledge is by definition implicit and does not lend itself to the kinds of techniques used in traditional cognitive modeling approaches (e.g., think-alouds, interviews). We presented and evaluated two similarity learning approaches that can determine which visual features students attend to when perceiving visual representations.

9. ACKNOWLEDGMENTS

We thank Professor John Moore for his help in recruiting participants for this study, and the LUCID group for their suggestions.

10. REFERENCES

[1] Ainsworth, S.: 'The educational value of multiple-representations when learning complex scientific concepts',

in Gilbert, J. et al. (Eds.): 'Visualization' (Springer, 2008), pp. 191-208

[2] NRC: 'Learning to think spatially' (National Academies Press, 2006)

[3] Wertsch, J., & Kazak, S.: 'Saying more than you know in instructional settings', in Koschmann, T. (Ed.): 'Theories of Learning and Studies of Instructional Practice' (Springer, 2011), pp. 153-166

[4] Airey, J., & Linder, C.: 'A disciplinary discourse perspective on university science learning', J. of Research in Science Teaching, 2009, 46, pp. 27-49

[5] Dreher, A., & Kuntze, S.: 'Teachers facing the dilemma of multiple representations being aid and obstacle for learning', Journal für Mathematik-Didaktik, 2014, pp. 1-22

[6] Ainsworth, S.: 'DeFT: A conceptual framework for considering learning with multiple representations.', Learning and Instruction, 2006, 16, pp. 183-198

[7] Gilbert, J.: 'Visualization: A metacognitive skill in science and science education', in Gilbert, J.K. (Ed.): 'Visualization in science education' (Springer, 2005), pp. 9-27

[8] Schnotz, W.: 'An integrated model of text and picture comprehension', in Mayer, R.E. (Ed.): 'The Cambridge Handbook of Multimedia Learning' (Cambridge University Press, 2005), pp. 49-69

[9] Koedinger, K., & Corbett, A.: 'Cognitive Tutors: Technology bringing learning sciences to the classroom', in Sawyer, R. (Ed.): 'Cambridge Handbook of the Learning Sciences ' (Cambridge University Press, 2006), pp. 61-77

[10] VanLehn, K.: 'The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems', Educational Psychologist, 2011, 46, (4), pp. 197-221

[11] Tuckey, H., Selvaratnam, M., & Bradley, J.: 'Identification and rectification of student difficulties concerning three-dimensional structures, rotation, and reflection', J. of Chemical Education, 1991, 68, pp. 460-464

[12] Davidowitz, B., & Chittleborough, G.: 'Linking the macroscopic and sub-microscopic levels: Diagrams', in Gilbert, J., and Treagust, D. (Eds.): 'Multiple representations in chemical education' (Springer, 2009), pp. 169-191

[13] Seufert, T.: 'Supporting coherence formation in learning from multiple representations', Learning and Instruction, 2003, 13, pp. 227-237

[14] Kellman, P.J., & Massey, C.M.: 'Perceptual Learning, cognition, and expertise', The Psychology of Learning and Motivation, 2013, 558, pp. 117-165

[15] Massey, C., Kellman, P., Roth, Z., & Burke, T.: 'Perceptual learning and adaptive learning technology', in Stein, N., and Raudenbush, S. (Eds.): 'Developmental cognitive science goes to school' (Routledge, 2011), pp. 235-249

[16] Kellman, P., Massey, C., & Son, J.: 'Perceptual learning modules in mathematics.': 'Topics in Cognitive Science' (2009), pp. 285-305

[17] Anderson, J., Boyle, C., Corbett, A., Lewis, M.: 'Cognitive modeling and intelligent tutoring' (MIT Press, 1990)

[18] Rau, M., Aleven, V., Rummel, N., & Rohrbach, S.: 'Why interactive learning environments can have it all: Resolving design conflicts between conflicting goals': 'Proceedings of SIGCHI 2013' (ACM, 2013), pp. 109-118

[19] Clark, R., Feldon, D., Van Merrienboër, J., Yates, K., & Early, S.: 'Cognitive task analysis', Spector, J. et al. (Eds.): 'Handbook of research on educational communications and technology' (Lawrence Erlbaum, 2007), pp. 577-593

[20] Schraagen, J., Chipman, S., & Shalin, V.: 'Cognitive Task Analysis' (Erlbaum Associates, 2000)

[21] Koedinger, K., Corbett, A., & Perfetti, C.: 'The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning', Cognitive Science, 2012, 36, pp. 757–798

[22] Rappoport, L., & Ashkenazi, G.: 'Connecting levels of representation: Emergent versus submergent perspective', Int. J. of Science Education, 2008, 30, (12), pp. 1585-1603

[23] Talanquer, V.: 'On cognitive constraints and learning progressions: The case of "structure of matter"', Int. J. of Science Education, 2009, 31, (15), pp. 2123-2136

[24] Goldstone, R., Landy, D., & Son, J.: 'The education of perception', Topics in Cognitive Science, 2010, 2, pp. 265-284

[25] Rau, M.: 'Multi-methods approach for domain-specific grounding: An ITS for connection making in chemistry', Under review at IEEE TLT.

[26] Jamieson, K., Jain, L., Fernandez, C., Glattard, N., & Nowak, R.: 'NEXT: A system for real-world development, evaluation and application of active learning': 'Advances in Neural Information Processing Systems' (2015), pp. 2638-2646

[27] Stewart, N., Brown, G., & Chater, N.: 'Absolute identification by relative judgment', Psychological Review, 2005, 112, pp. 881–911

[28] Bijmolt, T., & Wedel, M.: 'Effects of alternative methods of collecting similarity data for multidimen-sional scaling', Int. J. of Research in Marketing, 1995, 12, pp. 363–371

[29] Agarwal, S., Wills, J., Cayton, L., Lanckriet, G., Kriegman, D., & Belongie, S.: 'Generalized non-metric multidimensional scaling': 'Proceedings of the 12th Int. Conference on Artificial Intelligence and Statistics' (2007)

[30] Johnson, R.: 'Pairwise nonmetric multidimensional scaling', Psychometrika, 1973, 38, (1), pp. 11–18

[31] Tamuz, O., Liu, C., Belongie, S., Shamir, O., & Kalai, A.: 'Adaptively learning the crowd kernel': 'Proceedings of the 28th Int. Conference on Machine Learning' (2011)

[32] Chechik, G., Sharma, V., Shalit, U., & Bengio, S.: 'Large scale online learning of image similarity through ranking', Journal of Machine Learning research, 2010, pp. 1109–1135

[33] Atzmon, Y., Shalit, U., & Chechik, G.: 'Learning sparse metrics, one feature at a time', Journal of Machine Learning Research, 2015, 1, pp. 1-48

[34] Arias-Castro, E.: 'Some theory for ordinal embedding', arXiv preprint arXiv:1501.02861, 2015

[35] Negahban, S., Oh, S., & Shah, D.: 'Iterative ranking from pair-wise comparisons': 'Advances in Neural Information Processing Systems' (2012), pp. 2474-2482

[36] Yuan, M., & Lin, Y.: 'Model selection and estimation in regression with grouped variable', J. of the Royal Statistical Society: Series B, 2006, 68, (1), pp. 49–67

Student Usage Predicts Treatment Effect Heterogeneity in the Cognitive Tutor Algebra I Program

Adam C Sales
University of Texas
College of Education
Austin, TX, USA
asales@utexas.edu

Asa Wilks
RAND Corporation
Santa Monica, CA, USA
awilks@rand.org

John F Pane
RAND Corporation
Pittsburgh, PA, USA
jpane@rand.org

ABSTRACT

The Cognitive Tutor Algebra I (CTAI) curriculum, which includes both textbook and online components, has been shown to boost student learning by about 0.2 standard deviations in a randomized effectiveness trial. Students who were assigned to the experimental condition varied substantially in how, and how much, the used the online component of CTAI, but original analyses of the experimental data focused on estimating average effects, and did not examine whether the CTAI treatment effect varied by the amount of style of usage. This study leverages log data from the experiment to present a more nuanced analysis. It uses the framework of Principal Stratification, which estimates the varying CTAI treatment effect as a function of "potential" usage—either how students used the program, or how they would have used it had they been assigned to the treatment condition. With experimental data, Principal Stratification does not require that we assume that all relevant variables have been measured. With this framework, we find that students who receive a medium amount of assistance from the software (in the form of hints and error feedback) experience the largest effects, with lower effects for students who receive a lot or a little; and evidence that students who do not follow the curriculum order experience smaller treatment effects.

Keywords

Causal Mechanisms, Principal Stratification, Intelligent Tutors, Bayesian Hierarchical Models

1. INTRODUCTION

Intelligent tutors—computer programs designed to teach—claim to improve student achievement via a number of mechanisms, including a reliance on cognitive modeling, instant feedback, and individualized instruction. As the demand for intelligent tutors grows, so does the demand for evidence of their effectiveness, and the educational research community has kept apace, with a number of randomized field trials [e.g. 5, 9, 14]. Since intelligent tutors are computerized, it is relatively easy for experimenters to collect student log data, alongside traditional evaluation data. This paper will provide a template for how to evaluate the log data from an intelligent tutor experiment, to help elucidate the intelligent tutors' mechanisms and when and for whom they work.

A recent randomized study of Carnegie Learning's Cognitive Tutor Algebra I (CTAI) curriculum, under real-life conditions, was reported in [8]. In the second year of the experiment, in high school classrooms, the study found, that CTAI boosts student learning by about 0.2 standard deviation, on average. However, in the first year of the experiment CTAI's effect was close to nil. Surely one explanation for this heterogeneity is that students and teachers used the curriculum differently in the two years—but how? What aspects of student usage predict a treatment effect?

The effectiveness trial produced extensive student usage data, as the computer program logged students' activity. In this paper, we use this data—in particular, usage data from the 2nd-year high school sample that apparently experienced a substantial CTAI effect—to explore the relationship between student usage and causal effects. In future work, we will attempt to use these findings to explain the difference between the two years of the experiment.

A preliminary study, [17], argued that the best causal model for the usage data relies on the "principal stratification" framework [2, 7], under which students who used the CTAI software in a particular way are compared to control students who would have used it in the same way, had they been assigned to treatment. This study is the first full study that last year's preliminary study promised. It provides two sets of results exploring different aspects of CTAI's mechanisms: an analysis of assistance, which is calculated from the hints that students request and the errors they make, and an analysis of the the order at which students work on CTAI's sections. The paper also includes a more detailed discussion of the models, and a discussion of some issues with the results in [17].

2. DEFINING THE QUESTION: HOW DOES POTENTIAL USAGE MODERATE THE CTAI EFFECT?

As in [17], in this paper we model student usage under the principal stratification (PS) framework, a generalization of the Neyman-Rubin Causal Model [15] of potential outcomes. If Z is a binary treatment assignment, and Y

is an outcome, each subject has two potential outcomes: $Y(Z = 1)$ and $Y(Z = 0)$, the outcome she would present under the treatment condition, and under the control condition, respectively. Each of these is defined, though unobserved, prior to treatment assignment Z. After subjects have been assigned to treatment, exactly one of the potential outcomes is observable for each subject: for treatment subjects, the observed $Y = Y(Z = 1)$, and for control subjects, $Y = Y(Z = 0)$.

[2] generalized the potential outcomes framework, introducing the concept of principal strata. A principal stratum is a grouping of subjects based on potential values of intermediate outcomes. For example, if we call students' usage values U, each student has usage values $U(Z = 1)$ and $U(Z = 0)$—the usage they would exhibit under the treatment and control conditions, respectively. In the CTAI experiment, $U(Z = 0) = 0$ for all subjects, since no control subjects had access to the cognitive tutor. Say we model usage as a categorical value for K categories, $U = 1, ..., K$. Then there are k principal strata: $\{U(Z = 1) = k, U(Z = 0) = 0\}$ for $k = 1, ..., K$. In this framework, principal stratum membership is observed for students in the treatment group—we observe their usage once they are assigned to treatment, and we know from the experimental design that they would not have used the tutor had they been assigned to control. The potential usage for students in the control group, however, is unobserved, and must be estimated; the following section will discuss this process in more detail.

For each stratum, we can define a "principal effect": the average treatment effect $\tau_k = \mathbf{E}[Y(Z = 1) - Y(Z = 0)|U(Z = 1) = k, U(Z = 0) = 0]$ for subjects in principal stratum k. Although unobserved, these strata are defined prior to treatment assignment—if assigned to treatment, what *would* a student's usage be? That is, observed usage U is an intermediate outcome, or a mediator, but potential usage $U(Z = 0)$ and $U(Z = 1)$ is a pre-treatment covariate, or a moderator. The principal effects are, then, subgroup effects, for various levels of potential usage. Differences between principal effects are differences in the effect of CTAI for students who use (or would use) CTAI differently. To put it more precisely, consider the difference $\tau_j - \tau_k$. This is the difference in the effect of CTAI between the group of subjects who, if given the opportunity, would exhibit usage in the amount of j or the amount of k. While the effect estimates τ_j and τ_k are themselves causal (due to randomization) the difference between them could be due to the effect of usage, or to pre-treatment differences between students in the two groups. In other words, since usage values were not assigned randomly, the difference in CTAI effect between two usage principal strata are not necessarily causal. Still, estimating principal effects, and their differences, along with differences in the composition of principal strata, can shed light on the mechanisms of CTAI.

In one of our analyses below, usage is measured as a continuous, not categorical, variable, so the PS approach entails discretizing usage scores. [4] suggested an alternative: modeling potential usage as a continuous mediator, via an interaction in a regression analysis. They refer to this analysis as a "causal effect predictiveness" or CEP curve. CEP curves are directly analogous to principal strata effects, but with continuous intermediate variables.

3. ESTIMATING PRINCIPAL EFFECTS AND CEP CURVES

Estimating principal effects and CEP curves is a complex process, since first we must estimate unobserved principal strata membership or potential usage variables, and only then to estimate treatment effects. In fact, principal effects, in some circumstances, are only partially identified—even in an infinite sample, a Bayesian credible interval for a principal effect may have a finite width. This is especially the case when researchers attempt to estimate principal effects without covariates, and while relaxing traditional instrumental variables assumptions. However, in the presence of covariates that predict usage variables, we may estimate informative effects.

This section describes the models that we use to estimate principal effects and CEP curves. More details can be found in [16].

3.1 The Model

In general, the central challenge in PS modeling is that principal strata membership is unknown. In the CTAI experiment, since control students had no access to CTAI software, strata membership for the treatment group is known, but must be estimated for the control group. The distribution of the potential outcomes for Y, conditional on covariates, $p(Y(Z = 0)|X_i)$, can be decomposed into the probability distribution of Y given $U_i(Z = 1)$, which is the distribution of interest, times the distribution of $p(U_i(Z = 1)|X_i)$, which, due to random assignment, may be estimated from the treatment group. Then, we may estimate the parameters of $p(Y(Z = 0)|U(Z = 1) = a, X)$ and compare them to the analogous distribution $p(Y(Z = 1)|U(Z = 1) = a, X)$ yielding estimates of treatment effects within principal strata.

If we assume that outcomes are conditionally normally distributed, the result is a finite normal mixture model:

$$p(Y_i(Z = 0)|X_i) = \sum_{k=1}^{K} Pr(U_i(Z = 1) = k|X_i)\phi(\mu_k(Z = 0) + f_k(X_i), \sigma_k) \quad (1)$$

and

$$p(Y_i(Z = 1)|X_i, U(Z = 1) = k) = \phi(\mu_k(Z = 1) + f_k(X_i), \sigma_k) \quad (2)$$

where $\phi(\mu, \sigma)$ is the normal density with mean μ and standard deviation σ. Equations (1)-(2) additionally assume no interaction between covariates and treatment status within principal strata. The contribution of covariates X_i to the mean of $Y_i(Z = 1)$ can vary from stratum to stratum, but within stratum it does not vary with treatment status. In practice, we estimate $f_k(X_i)$ as linear in covariates:

$$f_k(X_i) = X_i^T \beta_k \quad (3)$$

where we estimate a different set of slopes β in each stratum k. The linearity assumption can be relaxed or adjusted based on the model's fit to the data. The effect of CTAI in the k^{th} principal stratum is $\tau_k = \mu_k(Z = 1) - \mu_k(Z = 0)$.

The model to estimate a CEP curve is broadly similar to the PS model, with one important difference. In the PS model, usage was parametrized as a categorical variable, and different effects were calculated for each stratum. In the CEP framework, usage is continuous, and its interaction with the effect of treatment must be modeled. As the next section will discuss, we chose to model the CTAI effect as quadratic in usage, for instance. The CEP outcome model, then, is

$$p(Y_i(Z=0)|X_i) = p_{U(Z=1)|X_i}(a)\phi(f_{U|Z=0}(a) + f_X(X_i), \sigma). \quad (4)$$

and

$$p(Y_i(Z=1)|X_i, U(Z=1)=a) = \phi(f_{U|Z=1}(a) + f_X(X_i), \sigma). \quad (5)$$

where $p_{U(Z=1)|X}(a)$ is the density of $U(Z=1)$ conditional on X, $f_{U|Z=0}(a)$ and $f_{U|Z=1}(a)$ are parametric functions of usage for treated and untreated subjects, respectively, and $f_X(X_i)$ is a model for covariates. The CTAI treatment effect is now a function of potential usage, $U(Z=1)$: $\tau(a) = f_{U|Z=1}(a) - f_{U|Z=0}(a)$.

Models (1), (2), (4), and (5) all require a model for the density of usage, as a function of covariates X. In our paper, the usage model, $p(U(Z=1)|X)$, is also linear in X. When the the usage variable is continuous, it is:

$$p(U(Z=1)|X) = \phi(X\gamma, \sigma_U) \quad (6)$$

normal-theory linear regression. In PS models, when we discretize U, we do so *after* fitting model 6.

When U is binary, we use a linear logistic regression to estimate $p(U(Z=1)|X)$:

$$Pr(U(Z=1)|X) = invLogit(X\gamma) \quad (7)$$

We fit all of the above models simultaneously with Markov Chain Monte Carlo (MCMC), using JAGS and R [10, 11]. Since MCMC is a Bayesian technique, it required priors; we put a normal prior with mean zero and standard deviation 3 on each of the model fixed effects—a prior that easily accommodates any plausible effect, but discourages outlandish estimates. We put a weakly-informative inverse-gamma(0.001, 0.001) prior on the variance parameters.

The models for assistance, described below in Section 5, were fit with the Stampede Supercomputer at the Texas Advanced Computing Center.

3.2 Some Potential Pitfalls

[17] presented a set of preliminary results from principal stratification analyses. They were presented as a first attempt at fitting principal stratification models, to illustrate the technique and its potential for helping us understand some of the factors behind CTAI's effect. However, since the EDM 2015 conference, a number of issues emerged with the preliminary results in that paper. It is instructive to discuss those results as an illustration of potential pitfalls in principal stratification analysis.

3.2.1 Model Convergence

One of the first checks of a Markov Chain Monte Carlo model is convergence. MCMC models (ideally) proceed through two stages: first, in the "burn-in" stage, parameter estimates fluctuate widely as the model converges on the posterior distribution for the parameters. After convergence, the algorithm draws from the posterior distribution of the parameters. From these draws, we can estimate the posterior's mean—a point estimate for the parameters—standard deviation, and quantiles. However, it is not always clear when the burn-in period has ended, and the model has begun sampling from the posterior. There are two principal ways of checking this. Both methods rely on running the MCMC separately in two or more chains. That is, start the Gibbs sampler c separate times, with c sets of starting values for the parameters, and let the c separate chains each take their own course. Then, the results from the c chains may be compared; if the model has converged, they should resemble one another, since they each would have converged on the true posterior distribution. One method of measuring whether this is the case is the Gelman-Rubin R-hat statistic, which compares the within-chain variance two the between-chain variance; since, after the burn-in stage, the chains should all be sampling from the same distribution, the between-chain variance should be small. At convergence, the R-hat statistic should be approximately one. Typically, values of R-hat less than 1.1 are acceptable. Additionally, analysts may inspect "traceplots": plots of the c chains for each parameter. If the chains are each stationary—that is, not changing in location or variance—and seem to share a location and scale with each other, the model has most likely converged. If the various chains converge on different distributions, the model might be non-identified, or multi-modal—several different estimates might be equally consistent with the data.

Some of the models in [17] may not have achieved convergence. In this paper, all of the models had clearly achieved convergence.

3.2.2 Gain-Score Modeling and Covariate Selection

A second concern with the model results from [17] emerged from our use of gain-scores—the difference between a post-test and a pre-test—as the outcome in the model, as opposed to the post-tests themselves. The problem with doing so is that the usage model was linear in the pre-test, by design. In the assistance model, for instance, assistance is anti-correlated with pretests, so the the control subjects who were estimated to have high levels of potential assistance also had high pre-test scores. On the other hand, pre-test scores are anti-correlated with gain scores, due to regression to the mean. So the control subjects with high estimated assistance will have lower gain scores on average. This can lead to an overestimate of an effect in the high-assistance stratum, especially if the usage model is misspecified. In principle this is an easy problem to correct, simply by including pre-test scores as a covariate in the outcome model as well. However, doing so would undermine the rationale of gain score modeling. For these reasons, we relied exclusively on post-test modeling in this paper, with the pre-test as a covariate in both the usage and outcome sub-models.

3.2.3 Student-Level Averages as Usage Variables

[17], and an earlier version of this manuscript, estimated the variation of the CTAI effect as a function of the av-

erage number of hints and errors each student requested or committed (called "assistance").[1] These averages were taken over all of each student's worked problems. Subsequent analysis revealed a curious phenomenon: the students with the most extreme average assistance values worked very few problems–almost uniformly so. Interpreting the CEP curve, in this case, becomes nearly impossible, since average assistance is so closely related to the amount of usage. The reason for the close relationship is straightforward: sample averages are random variables, and the variance of a sample mean is directly proportional to the sample size. The average assistance values for the group of students who worked very few problems had a high variance; conversely, the variance of average assistance for students who worked a large number of problems was much smaller.

The solution we chose for this issue was to run the model not on student-level average assistance values, but on problem-level data directly, adding another level into the multilevel structure. That way, the model considers student-level usage variables to be latent, as opposed to manifest (i.e. directly observed). Extreme values of latent variables estimated from a small number of problems enter into the model less as students with extreme usage patterns, and more as students whose usage is poorly-determined. In other words, from one MCMC draw to another, the estimate for each low-usage student's assistance value would vary considerably, so low-usage students would contribute little to the overall estimate of the CEP curve. We discuss the problem-level assistance model in Section 5.

3.2.4 Model Validation

The difficulty of constructing correct principle stratification models, and the ease of constructing models that yield misleading results, suggests that PS models should undergo rigorous specification checking before they are believed. [1], an excellent example of careful principal stratification analysis, provided guidance on how to validate a PS model, which we followed. We conducted three types of checks with each model:

- Estimating each effect with multiple different models and checking for concordance. In the assistance analysis, we estimated MCMC models treating the usage variable as either categorical or continuous. In both analyses we estimated both a normal-distribution model, as discussed in in Section 3.1, and a "robust" model, in which we substituted student's t-distribution for normal distributions in the model, allowing for outliers.
- Inspecting residual plots to assess model fit, for both the usage model and the outcome model.
- Estimating models with made-up outcome data. We did this primarily with a placebo outcome, generated by adding random noise to the pre-test variable. We then hoped not to find any treatment effects.

[1]The original manuscript also included an analysis of each student's average number of problems per section, which fell prey to the same issues as the assistance analysis. We will revisit the problem-per-section analysis in future work.

In this paper, due to space constraints, we included estimates from alternative methods, but not residual plots or placebo results; these, though, are available upon request.

Unfortunately, we cannot claim, at this point, that a method or model exists that will always recover the correct answer and never mislead—each model needs to be carefully tailored to its data, and then validated.

4. THE DATA

The CTAI experiment is described in [8]. The study was conducted in 73 high schools and 74 middle schools in 52 urban, suburban, and rural school districts in seven states, encompassing nearly 18,700 high school students and 6,800 middle school students. The schools were matched on a set of covariates prior to randomization, and were subsequently randomized to treatment or control conditions within matched pairs.

The study was an effectiveness trial, where the intervention must be adopted in as naturalistic conditions as possible. This means the study is supposed to capture common implementation variation resulting from imperfect implementation or even refusal to implement certain instructional materials. The naturalistic design of the experiment is particularly important for our analysis of student usage—usage patterns in the experiment plausibly correspond with what we may expect in general.

For the current study, we used only data from the second cohort in high schools. This is because that was the stratum in which overall effects were detected at the 5% level. Indeed, in the first year of implementation point estimates for the effect were close to zero. It may be the case that the difference in effect between the first and second years (a difference which itself is statistically significant) is due to different usage patterns. We hope that our larger project of estimating treatment effect heterogeneity by usage will help explicate the heterogeneity by cohort.

Software usage data is available for only a subset of the students in the treatment group. Considering only students who were present at post-test and are thus a part of outcomes analyses, we have usage logs for 83%. Students not present at post-test are considered to have attritted from the study.

The percentage of non-attritted students for whom we have usage data varies by school, from 0% (n=3 schools) to 100% (n=20 schools). We assume that schools that have 0% coverage did not implement the CTAI curriculum, despite being assigned to the treatment group. Carnegie Learning was unable, for technical reasons, to retrieve software usage log data for that school.

4.1 Imputing Missing Data

As described above, there were missing data values in the covariates, as well as in the student log scores. We used the `missForest` package in `R` [18, 11] to impute missing covariate values. The out-of-box normalized root mean-squared-error for the imputation was 0.02. Since this value is so low, since there was a relatively small amount of missing data, and since covariates play a merely predictive role in our analy-

sis, we assumed that the uncertainty from other aspects of the model would dominate the uncertainty due to covariate imputation and only imputed one dataset, rather than a full multiple imputation.

Missing usage data presents a more serious problem. First, some schools in the CTAI study were not included in the usage dataset. We deleted these schools from the analysis, along with their matched pairs. Since a matched randomized experiment is an aggregate of a randomized trial in each matched pair, discarding the matched pairs with missing data is nearly benign.

We classified within-school missing usage data into two groups: some students did not have usage data because they did not use the software. Since absolute software usage is driven primarily by teachers, we calculated the proportion of students with missing data for each teacher. If almost all of a teacher's students were missing from the usage dataset, we assumed that they did not use the tutor in their classroom.

The rest of the missing student usage data was due to our inability to match students to their records. We assumed that these data were missing at random [6]—that their missingness was ignorable conditional on their measured covariates. The missingness was likely not missing completely at random, since students who were difficult to match generally did not fill out their student information thoroughly, and thoroughness may correlate with post-test scores or usage patterns. The imputation strategy for these missing data points was identical to the imputation of unobserved potential usage for the control students. That is, the same model that estimated densities for usage variables for control students also estimated missing usage data for some treated students. The missing data strategy in this case was, therefore, either full-information maximum likelihood or MCMC, depending on the analysis.

5. HINTS AND ERRORS

5.1 Assistance Scores

	#Errors=0	#Errors>0	Sum
#Hints=0	0.42	0.34	0.76
#Hints>0	0.01	0.23	0.24
Sum	0.43	0.57	1.00

Table 1: The proportions of problems in our dataset in which students make at least one error or request at least one hint.

[12] defined assistance as the sum of the number of hints students request and the number of errors they make, which together represent the feedback CTAI gives the students. High assistance indicates that a student is struggling.

Hints and errors vary from problem to problem, from section to section, and from student to student. Table 1 shows the joint probability of requesting at least one hint and making at least one error in our dataset. In 58% of worked problems, the student requested at least one hint or one error. Further, hints and errors tend to accompany each other: in only 1% of worked problems the student requested a hint without making an error. In many problems, hints and errors occur

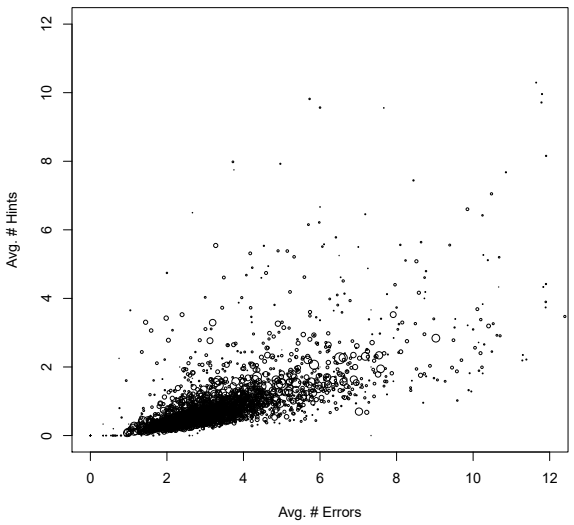


Figure 1: The average number of hints and errors requested for each student. The size of the plotted points is proportional to the square root of the number of problems they completed—and hence to the standard deviation of the plotted averages.

sequentially: a student will work part of the problem, perhaps make an error and receive feedback, perhaps request a hint, and then move on to the rest of the problem. It is important to keep in mind, then, that hints do not always precede errors—sometimes, they are the result of a prior error made while working the same problem.

Figure 1 plots the average number of hints a student requests as a function of the average number of errors he makes. While most students request between 0 and two hints per problem, and make between one and eight errors per problem, some students request far more hints or make far more errors. Further, students who request more hints are much more likely to make more errors. The size of the points in Figure 1 is proportional to the square root of the number of problems they completed—and hence to the standard deviation of the plotted averages. The extreme values in the figure typically come from students who work very few problems, as described in Section 3.2.3, complicating the interpretation of a model that uses average hints or errors as a mediator variable.

For that reason, we incorporated a problem-level sub-model for assistance into our larger principal stratification model. Rather than model the total number of hints and errors per problem, which would necessitate a complex, and possibly misspecified, count-data model, we modeled the probability of a student requesting a hint or making an error (or both) on each problem. The model was as follows:

$$Pr(A_{ip} \geq 1) = invLogit(U_i + \delta_{s[p]}) \quad (8)$$

Where A_{ip} is the total amount of assistance, i.e. hints and errors, that student i experiences from problem p. U_i is a

random student effect, representing the student's propensity to receive assistance on a problem, and $\delta_{s[p]}$ is a section random effect.[2]

The variable U_i, student i's "assistance score," is the mediator that we use to predict her CTAI treatment effect.

U_i is itself predicted, in turn, by a set of covariates including pretest scores, demographics, and teacher random effects nested within school random effects. The results of this usage model are available upon request. They show that prior test scores and "gifted" status are inversely correlated with assistance scores—higher performing students are less likely to make errors or request hints. Special education students are more likely to receive assistance, and males are less likely than females.

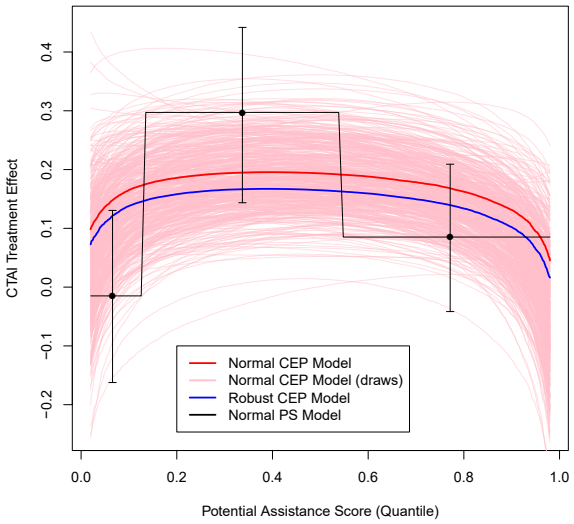


Figure 2: Assistance model results: $\mathbf{E}[Y(Z=1) - Y(Z=0)|U(Z=1)]$, CTAI treatment effect as a function of potential assistance $U(Z=1)$ quantiles. Results are shown for an MCMC CEP normal-distribution model that treats assistance as continuous, a "robust" CEP model based on the t-distribution that allows for outliers, and a normal-distribution PS model that breaks assistance scores into high, medium, and low categories. To display statistical uncertainty, we also plotted 500 draws for the effect function from the CEP model, and 95% credible intervals (error bars) for the three PS effects. The treatment effect is in effect size units.

Figure 2 shows the results for three models—a normal- distribution CEP model, a robust CEP model, and a normal-distribution PS model—which roughly agree that treatment effects are highest for students with assistance scores in the center of the distribution, and lower for students who used a high or low amount of assistance. The PS model, in which assistance scores were discretized, reports more exaggerated differences between treatment effects for students with medium assistance scores and those with high or low scores; these differences, moreover, are highly significant—the probabilities that the average effect for medium students is higher than that for low and high students are 1 and 0.987, respectively. However, when the estimation error is taken into account, it is apparent that the CEP and PS models do not necessarily disagree.

There are a number of ways to interpret these results. The results reflect varying CTAI effects for various usage patterns. One of CTAI's selling points is the instant feedback it provides students as they work through and complete problems. Students who under-utilize this service—in the low assistance stratum—are then likely to experience a smaller CTAI effect. This may be because they began as excellent students—assistance is anti-correlated with pretest scores—and hence did not need the extra help that CTAI provides. Alternatively, students with low assistance scores may be under-utilizing the service for a different reason; perhaps they feared that requesting too many hints, or making too many mistakes, would slow their progress through the tutor, so they were overly cautious.

Students who request hints or make errors quickly, without slow deliberation, may not be able to learn from the problems they work. Some students "game" the system, by requesting hints until they are provided with the correct answer, or they simply do not try very hard to figure out the answer themselves. It may be that the students in the CTAI experiment with very high assistance scores, experience lower treatment effects for some of these reasons Alternatively, they might have struggled with the material in general, and required more personalized help from a teacher, as opposed to a computerized tutor.

However, students in the middle of the assistance distribution experienced large CTAI effects, suggesting an assistance "sweet spot." In future trials, teachers could be instructed to encourage their students to use a medium number of hints, and complete problems with a moderate amount of caution—trying hard to answer problems correctly, but also allowing themselves to make mistakes. If this strategy leads to higher CTAI effects, it suggests that part of the CTAI effect heterogeneity across usage patterns is causal—that using the system differently leads to higher effects.

6. SKIPPING SECTIONS

An important part of the design of CTAI is the scaffolding of skills and knowledge. The skills that students learn in Algebra I build on each other, so the order in which students learn material and master skills matters—at least in theory. The design of CTAI accounts for this order, by insisting that students master certain skills before moving on to others. Indeed, that is the notion that lies behind the sections of the CTAI curriculum.

We attempted to test the hypothesis that this scaffolding matters—that is, do students who the CTAI curriculum learn more from CTAI than students who do not? To answer

[2]The conventional item response theory model in this case would have a problem effect instead of a section effect. We chose section effects rather than problem effects since there are 5438 problems (that only appear once in the dataset, making problem effects difficult to estimate.

this question, we compared the order in which students in the CTAI experiment worked on sections to the intended order. About 80% of students worked on the sections in order. However, 20% of students skipped at least one section. Did the students who skipped one or more sections experience the same CTAI effect as those who completed the sections in the intended order? More precisely, is the CTAI effect the same in the principal stratum of students who, if assigned to CTAI, would complete the section in order, and in the principal stratum of students who, if assigned to CTAI, would skip at least one section?

A complication in estimating counterfactual stratum membership for control students in this case was that in the CTAI setup, teachers, not students, control which sections the students work on. Indeed, there were 38 teachers in the treatment group for whom we had data on whether students skipped a section. Of those 38 teachers, 17 teachers did not have any students who skipped any sections at all, while there were five teachers more than 80% of whose students skipped sections. Since such a large proportion of the variation in section-skipping occurred at the teacher level, we included a set of teacher-level predictors in our usage model. An anonymous reviewer alerted us to the threat of over-fitting; hence, due to the small number of teachers in the treatment group, we chose only two teacher level covariates in the model: percent ESL, and average pre-test. The small covariate-to-sample size ratio at both the student and the teacher levels, combined with the informative priors [See 3], should alleviate concerns of over-fitting.

The usage model, whose results are available upon request, was unsuccessful in estimating precise effects for any covariate, but in aggregate was able to predict stratum membership. One exception is that students with higher pretest scores are more likely to skip sections, as are teachers whose students have higher pretest scores on average.

Stratum	Effect (Normal)	Effect (Robust)
Do Not Skip	0.27	0.19
	0.09	*0.07*
	(0.06–0.44)	(0.05–0.33)
Skip $\geq$ 1 If Treated	-0.09	-0.07
	0.13	*0.11*
	(-0.33,0.17)	(-0.28,0.48)
Difference	-0.36	-0.26
	0.12	*0.11*
	(-0.59,-0.12)	(-0.48,-0.03)

Table 2: The CTAI effect in the two principal strata defined by whether a not a student would skip a section if they were assigned to the treatment. We estimated principal effects with both an MCMC model based on the normal distribution, based on the more robust student's t-distribution. Standard deviations of the posteriors are in italics, and 95% credible intervals (MCMC) are provided in parentheses under the estimates.

The results of our analysis are in Table 2 and Figure 3. Both models detect significantly greater treatment effects in the principal stratum of students who would not skip sections if assigned to the treatment, than in the stratum of students who would. This might be taken as evidence that the order in which students complete sections plays a large role in the effectiveness of CTAI. Alternatively, it may be that teachers who tinker with the order of sections that their students work are likely to tinker with other aspects of the CTAI design as well, to deleterious effect (perhaps along the lines of [13]). In either reading, the effect of CTAI is not merely due to the practice it gives students, or immediate feedback, but also to its underlying pedagogical and cognitive theory.

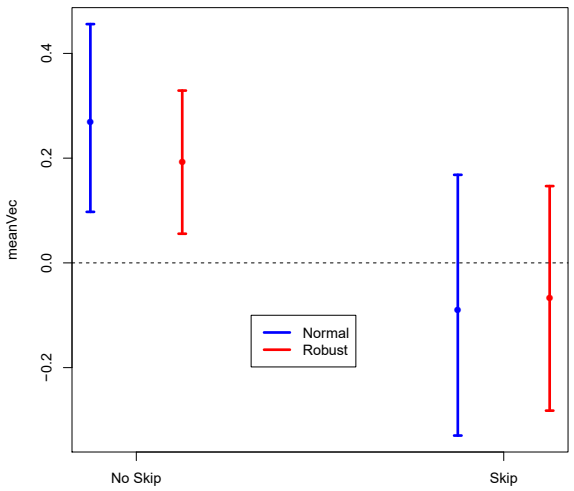


Figure 3: Estimates, and 95% credible intervals, for the CTAI effect in the principle stratum of students who would not skip sections, and in the stratum of students who would. The results plotted for both the normal and t-distribution ("robust") models.

A third possibility is that the entire difference is driven by an underlying teacher or student characteristic, such as ability; students with higher pretest scores are more likely to skip sections—perhaps the treatment effect is significantly lower for them, as well.

7. DISCUSSION

We showed that without additional identification assumptions, researchers can use log data to form a deeper understanding of their software's effect. However, we also discussed some of the difficulties in estimating these models correctly.

We updated and clarified a result from our preliminary study [17]. We find that the relationship between the amount of assistance students receive from CTAI and the CTAI treatment effect they experience is not monotonic. The highest effects appear for the students who receive a medium amount of assistance; those who receive much more or less experience smaller treatment effects, on average. This may be the result of student attributes—that the students at the margins are either too advanced or gaming the software—or it may be that certain modes of software usage are better than

others.

Next, we investigated if students who skip a section in the recommended curriculum, working on sections out of order, may experience lower effects. The result may confirm part of the motivating theory behind CTAI: that Algebra I skills build on each other, so the order at which students work on material can contribute or detract from their success.

Along those lines, we plan a number of future analyses. We hope to update the preliminary study's results that suggested that the CTAI treatment effect increases with the amount of usage, and to investigate the dependence of the CTAI effect on students' mastery of sections. Further along, we hope to discover and define interesting multivariate principal strata, perhaps as the result of a cluster analysis of the high-dimensional usage data.

Finally, after cultivating a more complete understanding of the usage patterns that lead to higher CTAI effects, we can explore treatment-effect heterogeneity. In particular, we may be able to answer why in the first year of implementation CTAI did not seem to boost test scores, but in the second year it did. Was differential usage to blame?

In the meantime, this paper uses rigorous causal methods to confirm some previous hypotheses about CTAI's causal mechanisms, and points a way forward for future work modeling usage variables in experimental designs.

8. ACKNOWLEDGMENTS

This work is supported by the United States National Science Foundation Grant #DRL-1420374 to the RAND Corporation and by the Institute of Education Sciences, U.S. Department of Education, through Grant R305B1000012 to Carnegie Mellon University. The opinions expressed are those of the authors and are not intended to represent views of the Institute or the U.S. Department of Education or the National Science Foundation. The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources that have contributed to the research results reported within this paper. `http://www.tacc.utexas.edu`. Thanks to Steve Fancsali, Steve Ritter, and Susan Berman for processing and delivering the CTAI usage data. Thanks to Brian Junker for helpful advice and guidance.

References

[1] A. Feller, T. Grindal, L. W. Miratrix, and L. Page. Compared to what? variation in the impact of early childhood education by alternative care-type settings. *Annals of Applied Statistics*, 2016. in press.

[2] C. E. Frangakis and D. B. Rubin. Principal stratification in causal inference. *Biometrics*, 58(1):21–29, 2002.

[3] A. Gelman, A. Jakulin, M. G. Pittau, and Y.-S. Su. A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics*, pages 1360–1383, 2008.

[4] P. B. Gilbert and M. G. Hudgens. Evaluating candidate principal surrogate endpoints. *Biometrics*, 64(4):1146–1154, 2008.

[5] J. B. Heppen, K. Walters, M. Clements, A.-M. Faria, C. Tobey, N. Sorensen, and K. Culp. Access to algebra i: The effects of online mathematics for grade 8 students. ncee 2012-4021. *National Center for Education Evaluation and Regional Assistance*, 2011.

[6] R. J. Little and D. B. Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[7] L. C. Page. Principal stratification as a framework for investigating mediational processes in experimental settings. *Journal of Research on Educational Effectiveness*, 5(3):215–244, 2012.

[8] J. F. Pane, B. A. Griffin, D. F. McCaffrey, and R. Karam. Effectiveness of cognitive tutor algebra i at scale. *Educational Evaluation and Policy Analysis*, 2013.

[9] J. F. Pane, D. F. McCaffrey, M. E. Slaughter, J. L. Steele, and G. S. Ikemoto. An experiment to evaluate the efficacy of cognitive tutor geometry. *Journal of Research on Educational Effectiveness*, 3(3):254–281, 2010.

[10] M. Plummer. *rjags: Bayesian Graphical Models using MCMC*, 2016. R package version 4-5.

[11] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[12] S. Ritter, A. Joshi, S. Fancsali, and T. Nixon. Predicting standardized test scores from cognitive tutor interactions. In *EDM*, pages 169–176, 2013.

[13] S. Ritter, M. Yudelson, S. E. Fancsali, and S. R. Berman. How mastery learning works at scale. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 71–79. ACM, 2016.

[14] J. Rochelle, M. Feng, N. Heffernan, and C. Mason. Preliminary findings from an efficacy study of online mathematics homework, 2015. Poster presented at an US Dept of Education, Institute for Educational Sciences meeting of investigators of funded projects.

[15] D. Rubin. Bayesian inference for causal effects: The role of randomization. *The Annals of Statistics*, pages 34–58, 1978.

[16] A. C. Sales and J. Pane. Modeling the treatment effect from educational technology as a function of student usage, 2016. Conference Paper for AEFP Annual Conference 2016.

[17] A. C. Sales and J. F. Pane. Exploring causal mechanisms in a randomized effectiveness trial of the cognitive tutor. In *EDM*, 2015.

[18] D. J. Stekhoven. Missforest: nonparametric missing value imputation using random forest. *Astrophysics Source Code Library*, 1:05011, 2015.

LIVELINET: A Multimodal Deep Recurrent Neural Network to Predict Liveliness in Educational Videos

Arjun Sharma
Xerox Research Centre India
Arjun.Sharma@xerox.com

Arijit Biswas
Xerox Research Centre India
Arijit.Biswas@xerox.com

Ankit Gandhi
Xerox Research Centre India
Ankit.Gandhi@xerox.com

Sonal Patil
Xerox Research Centre India
Sonal.Patil@xerox.com

Om Deshmukh
Xerox Research Centre India
Om.Deshmukh@xerox.com

ABSTRACT

Online educational videos have emerged as one of the most popular modes of learning in the recent years. Studies have shown that liveliness is highly correlated to engagement in educational videos. While previous work has focused on feature engineering to estimate liveliness and that too using only the acoustic information, in this paper we propose a technique called LIVELINET that combines audio and visual information to predict liveliness. First, a convolutional neural network is used to predict the visual setup, which in turn identifies the modalities (visual and/or audio) to be used for liveliness prediction. Second, we propose a novel method that uses multimodal deep recurrent neural networks to automatically estimate if an educational video is lively or not. On the StyleX dataset of 450 one-minute long educational video snippets, our approach shows an relative improvement of 7.6% and 1.9% compared to a multimodal baseline and a deep network baseline using only the audio information respectively.

Keywords

Liveliness, Educational Videos, Recurrent Neural Network, Deep Learning, LSTM, Engagement, Multimodal Analysis.

1. INTRODUCTION

The amount of freely available online educational videos has grown significantly over the last decade. Several recent studies [1, 2, 3] have demonstrated that when educational videos are not engaging, students tend to lose interest in the course content. This has led to recent research activity in speaking style analysis of educational videos. Authors in [4] used crowd-sourced descriptors of 100 video clips to identify various speaking-style dimensions such as liveliness, speaking rate, clarity, formality etc. that drive student engagement and demonstrated that liveliness plays the most significant role in video engagement. Using a set of acoustic features and LASSO regression, the authors also developed automatic methods to predict liveliness and speaking rate. The Authors in [5] analyze the prosodic variables in a corpus of eighteen oral presentations made by students of Technical English, all of whom were native speakers of Swedish. They found out that high pitch variation in speech is highly correlated with liveliness. Arsikere et al. [6] built a large scale educational video corpus called StyleX for engagement analysis and provided initial insights into the effect of various speaking-style dimensions on learner engagement. They also found out that liveliness is the most influential dimension in making a video engaging. In this paper, we propose a novel multimodal approach called LIVELINET that uses deep convolutional neural networks and deep recurrent neural networks to automatically identify if an educational video is lively or not.

A learner can typically perceive or judge the liveliness[1] of an educational video both through the visual and the auditory senses. A lecturer usually makes a video lively by using several visual actions such as hand movement, interactions with other objects (board/tablet/slides) and audio actions such as modulating voice intensity, varying speaking rate etc. In the proposed approach, both visual and audio information from an educational video are combined to automatically predict the liveliness of the video. Note that a given lecture can also be perceived as lively based on the contextual information (e.g., a historic anecdote) that the lecturer may intersperse within the technical content. We however don't address this dimension of liveliness in this work [2].

This paper is novel in three important aspects. First, the proposed approach is the first of its kind that combines audio and visual information to predict the liveliness in a video. Second, a convolutional neural network (CNN) is used to estimate the setup (e.g., lecturer sitting, standing, writing on a board etc.) of a video. Third, Long Short Term Memory (LSTM) based recurrent neural networks are trained to classify the liveliness of a video based on audio and visual features. The CNN output determines which of the audio and/or visual LSTM output should be combined for the liveliness prediction.

We observe that there is a lot of variation in what is being displayed in an educational video, e.g., slide/board, lecturer, both slide/board and lecturer, multiple video streams showing lecturer and slide etc.. These different visual setups usually indicate to what degree the audio and the visual information should be combined for predicting liveliness. For example, when the video feed only displays the slide or the board, the visual features do not play a critical role in determining liveliness. However, when the video is focussed on

[1] defined as "full of life and energy/active/animated" in dictionary

[2] Note that the human labelers who provided the ground truth for our database [6] were explicitly asked to ignore this aspect while rating the videos

the lecturer, the hand gestures, body postures, body movements etc. become critical, i.e., the visual component plays a significant role in making a video lively. Hence, we first identify the setup of a video using a CNN based classifier. Next, depending on the setup, we either use both audio and visual information or use only the audio information from a video for training/testing of the LSTM networks. We train two separate LSTM based classifiers, one each for audio and visual modalities, which take a temporal sequence of audio/visual features from a video clip as input and predict if the clip is lively or not. Finally, audio/visual features from a test video clip are forward-propagated through these LSTMs and their outputs are combined to obtain the final liveliness label.

We perform experiments on the StyleX dataset [6], and compare our approach with baselines that are based on visual, audio and combined audio-visual features. The proposed approach shows relative improvement of 7.6% and 1.9% with respect to a multimodal baseline and a deep network baseline using only the audio modality respectively.

2. RELATED WORK

In this section, we discuss the relevant prior art in deep learning and multimodal public speaking analysis in videos.

Deep Learning: Recently deep neural networks have been extensively used in computer vision, natural language processing and speech processing. LSTM [7], a Recurrent Neural Network (RNN) [8] architecture, has been extremely successful in temporal modelling and classification tasks such as handwriting recognition [9], action recognition [10], image and video captioning [11, 12, 13], speech recognition [14, 15] and machine translation [16]. CNNs have also been successfully used in many practical computer vision tasks such as image classification [17], action recognition [18], object detection [19, 20], semantic segmentation [21], object tracking [22] etc.. In this work, we use CNNs for visual setup classification and LSTMs for the temporal modelling of audio/visual features.

Multimodal Public Speaking Analysis: Due to the recent development of advanced sensor technologies, there has been significant progress in the analysis of public speaking scenarios. The proposed methods usually employ use of multiple modalities such as microphone, RGB camera, depth sensor, kinect sensor, Google glasses, body wearables, etc. and analyse the vocal behaviour, body language, attention, eye contact, facial expression of the speakers along with the engagement of the audiences [23, 24, 25, 26]. Gan et al. [23] proposed baseline methods to do the quantification of several above mentioned parameters by analysing the multi-sensor data. Nguyen et al. [24] and Echeverria et al. [25] used kinect sensors to recognize the bodily expressions, body posture, eye contact of the speaker and thereby, providing feedback to the speaker. Chen et al. [26] presented an automatic scoring model by using basic features for the assessment of public speaking skills. It must be noted that all these works rely significantly on the sensor data captured during the presentation for their prediction task and hence, they are not applicable to educational videos that are available online. Moreover, all these approaches use shallow and hand-crafted audio features along with the sensor data. On the contrary, our proposed method uses deep learning based automatic feature extraction method for both audio and visual modalities from the video, and predicts the liveliness.

To the best of authors' knowledge, this is the first approach that uses a deep multimodal approach for educational video analysis.

3. PROPOSED APPROACH

In this section, we describe the details of the proposed approach. We begin with the description of how a given video is modeled as a sequence of temporal events, followed by the visual setup classification algorithm. Next, we provide the details of the audio and visual feature extraction. Finally, the details of the proposed multimodal method for liveliness prediction is described. The pipeline of the proposed approach is shown in Figure 1. The input to the system is a fixed length video segment of 10 seconds during both training and testing (referred to as 10-second clips throughout the paper). For any educational video of arbitrary length, 10-second clips are extracted with 50% overlap between the adjacent clips and the overall video liveliness label is determined based on the majority voting. In Section 5.1 we provide further details regarding extraction of these 10-second clips from the Stylex dataset.

3.1 Video Temporal Sequencing

Each 10-second clip is modeled as a temporal sequence of smaller chunks. If the total number of chunks in a 10-second clip is T, then $\{v_1, v_2, ..., v_t, ..., v_T\}$ and $\{a_1, a_2, ..., a_t, ..., a_T\}$ represent the temporal sequence of visual and audio features corresponding to each 10-second clip respectively. Note that, v_t (Section 3.3) and a_t (Section 3.4) are input to the visual and audio LSTM at time instant t.

3.2 Visual Setup Classification

One of our objectives is to automatically determine if both audio and visual information are required for liveliness prediction. If a video displays only slide/board, the visual features are less likely to contribute to the liveliness. However, if the camera displays that the lecturer is in a sitting/standing posture or is interacting with the content, the visual features could significantly contribute to the video liveliness. Hence, we collect a training dataset and train a CNN to automatically estimate the setup of a video. We describe the definition of the labels, the data collection procedure and the details of the CNN training in the next three subsections.

3.2.1 Video Setup Label Definition

We define five different categories which cover almost all of the visual setups usually found in educational videos.

- **Content:** This category includes the scenarios where the video feed mainly displays the content such as a blackboard or a slide or a paper. Frames, where the hand of the lecturer and/or pens or pointers are also visible, are included in this category. However, the video clips belonging to this category should not include any portion of the lecturer's face. Since the lecturer is not visible in this case, only the audio modality will be used for liveliness prediction.
- **Person Walking/Standing:** In this scenario, the content such as blackboard/slide are not visible. However, the lecturer walks around or remain in a standing posture. The lecturer's face and upper body parts (hand/shoulder) should be visible. Both audio and visual modality are used to predict liveliness in this case.
- **Person Sitting:** The content is not visible and the camera should focus only on the lecturer in a sitting posture. Both audio and visual modalities are considered for liveliness prediction.
- **Content & Person:** This includes all the scenarios where the upper body of the lecturer and the content both are visible. Frames, where the lecturer points to the slide/board or writes something on the board, are included in this category. Here also both the modalities are used for liveliness.

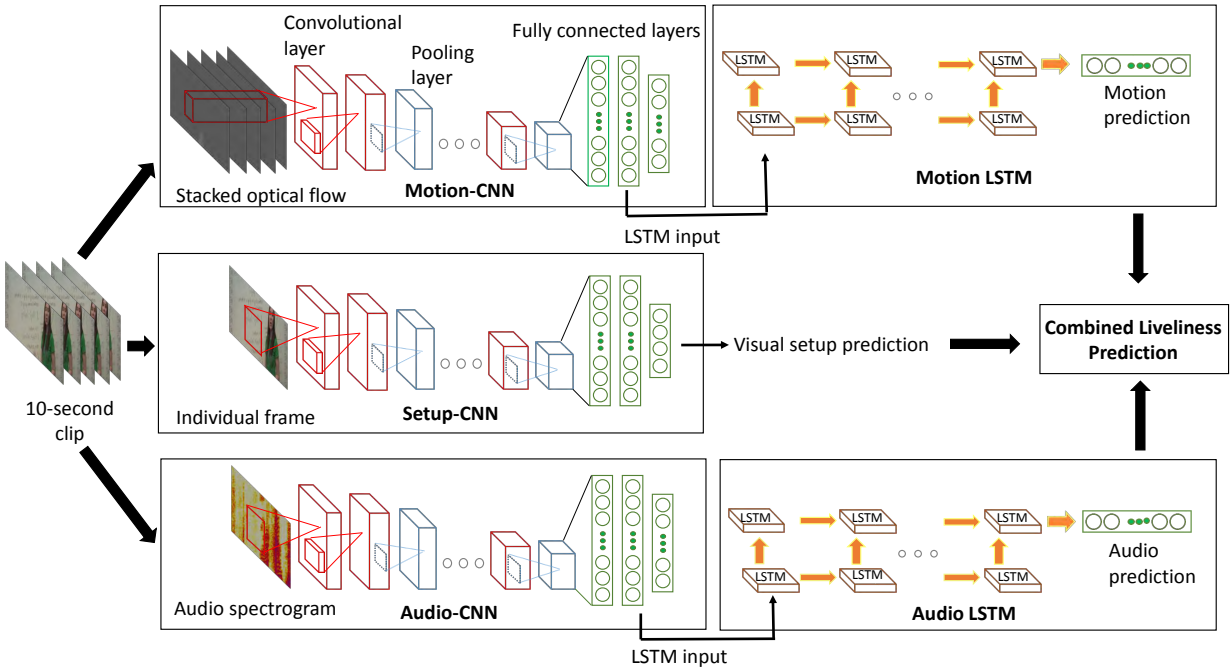


Figure 1: The overall pipeline of the proposed approach LIVELINET. The input to the system is a 10-second clip and output is the liveliness prediction label.

- **Miscellaneous:** This category includes all other scenarios which are not covered in the above four categories, e.g., two different video feeds for professor and content, students are also visible, multiple people (laboratory setups) are visible in the scene etc.. Since the frames from this category have significant intra-class variation and noise, we use only the audio information for liveliness prediction.

Some example frames from the above five categories are shown in Figure 2. The intra-class variation clearly shows the inherent difficulty of the setup classification task.

3.2.2 *Label Collection*

We used the StyleX dataset [6] for the liveliness prediction task. Although the liveliness labels were available along with the videos, video setup labels were not available. So we collect these additional labels using Amazon Mechanical Turk. We asked the Mturkers to look at the 10-second clips from StyleX and choose one of the five labels defined above. Each video clip is shown to three MTurk labellers and we assign the labels where at least two of the three labellers agreed. Although in most of the clips, all frames belong to only one of the above five categories, there were some 10-second clips (around 5%) where frames from more than one categories were present. In those cases, labellers were asked to provide the label based on the label of the majority of frames.

3.2.3 *CNN for Label Classification*

We used a CNN architecture to classify the setup of a 10-second clip. During training phase, all the frames belonging to a 10-second clip are used as the samples for the corresponding clip category. For this task, we use the same CNN architecture as used in [17]. In [17], the authors proposed a novel neural network model called Alexnet which improved the state-of-the-art imagenet classification [27] accuracy by a significant margin. Researchers in the computer vision community have often used the Alexnet architecture for other kinds of computer vision applications [28, 29]. Deep neural networks usually have millions of parameters. If the available training data for a particular classification task is not large enough, then training a deep neural network from scratch might lead to over fitting. Hence, it is a common practice to use a CNN which is already pretrained for a related task and fine-tune only the top few layers of the network for the actual classification task.

We fine-tune the final three fully connected layers (fc6, fc7, fc8) of Alexnet for visual setup classification. First, we remove the 1000 node final layer fc8 (used to classify 1000 classes form imagenet [17]) from the network and add a layer with only five nodes because our objective is to classify each frame into one of the five setup categories. Since, the weights of this layer are learned from scratch we begin with a higher learning rate of 0.01 (same as Alexnet). We also fine tune the previous two fully connected layers (fc6 and fc7). However, their weights are not learned from scratch. We use a learning rate of 0.001 for these layers while performing the gradient descent with the setup classification training data. Once the Alexnet has been fine-tuned a new frame can be forward propagated through this network to find the classification label. For a test 10-second clip, we determine the setup label for each frame individually and assign the majority label to the full clip. We refer to this CNN as Setup-CNN.

3.3 Visual Feature Extraction

In this section, we describe the details of the visual features used for predicting the liveliness of a video clip. The visual modality is

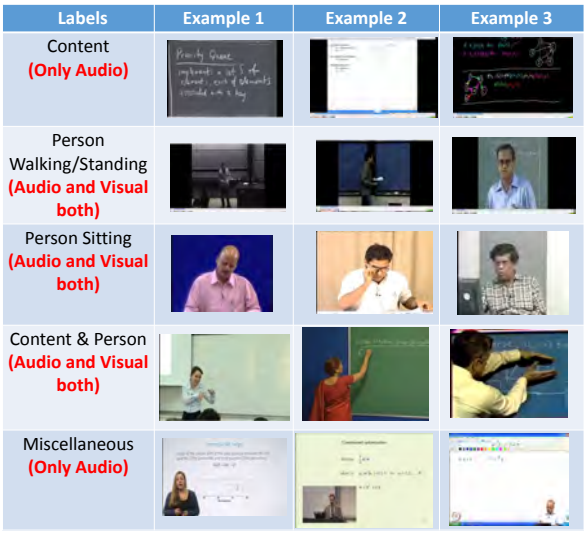


Figure 2: Example frames from different visual setup categories. We also point out the modalities which are used for liveliness in each of these setups.

used to capture the movement of the lecturer. We used a state-of-the-art deep CNN architecture to represent the visual information in the form of motion across the frames. Unlike the CNN model used in Section 3.2.3 (where input to the model was an RGB image comprising of 3 channels), the input to the CNN model in this section is formed by stacking horizontal and vertical optical flow images from 10 consecutive frames of a video clip. We refer to this CNN model as Motion-CNN in the subsequent sections of the paper.

For the Motion-CNN, we fine-tuned the VGG-16 temporal-net trained on UCF-101 [30] action dataset. The final fully connected layers (fc6, fc7, and fc8) of VGG-16 are fine-tuned with respect to the liveliness labels of the videos. The activations of the fc7 layer are extracted as the visual representation of the stacked optical flows which were provided as the input to the model. Given a 10-second clip, we generate a feature representation v_t (Section 3.1) from the corresponding 10 frame optical flow stack. We provide v_t as an input to LSTM module at time t to create a single visual representation for the full 10-second clip (Section 5.2).

Implementation Details: We use the GPU implementation of TVL1 optical flow algorithm [31]. We stack the optical flows in a 10-frame window of a video clip to receive a 20-channel optical flow image as an input (one horizontal channel and one vertical channel for each frame pair) to the Motion-CNN model. In Motion-CNN model, we also change the number of neurons in fc7 layer from 4096 to 512 before finetuning the model to get a lower dimensional representation of the 10 frame optical flow stack. We adopt a dropout ratio of 0.8 and set the initial learning rate to 0.001 for fc6, and to 0.01 for fc7 and fc8 layers. The learning rate is reduced by a factor of 10 after every 3000 iterations.

3.4 Audio Feature Extraction

We extract the audio feature a_t (Section 3.1) using a convolutional neural network. For each t, we find a corresponding one second long audio signal from the 10-second clip. We apply the Short-Time Fourier Transformation to convert each one second 1-d audio signal into a 2-D image (namely log-compressed mel-spectrograms with 128 components) with the horizontal axis and vertical axis being time-scale and frequency-scale respectively. The CNN features are extracted from these spectrogram images and used as inputs to the LSTM. We finetune the final three layers of Alexnet [17] to learn the spectrogram CNN features. We change the number of nodes in fc7 to 512 and use the fc7 representation corresponding to each spectrogram image as input to the LSTMs. The fine tuned Alexnet for the spectrogram feature extraction is referred as Audio-CNN. Learning rate and dropout parameters are chosen same as mentioned in Section 3.3.

3.5 Long Short Term Memory Networks

The Motion-CNN (Section 3.3) and the audio-CNN (Section 3.4) model only the short-term local motion and audio patterns in the video respectively. We further employ LSTMs to capture long-term temporal patterns/dependencies in the video. LSTMs map the arbitrary length sequential information of input data to output labels with multiple hidden units. Each of the units has built-in memory cell which controls the in-flow, out-flow, and accumulation of information over time with the help of several non-linear gate units. We provide a detailed description of LSTM networks below.

RNNs [8] are a special class of artificial neural networks, where cyclic connections are also allowed. These connections allow the networks to maintain a memory of the previous inputs, making them suitable for modeling sequential data. Given an input sequence $\mathbf{x}$ of length T, the fixed length hidden state or memory of an RNN $\mathbf{h}$ is given by

$$h_t = g(x_t, h_{t-1}) \quad t = 1, \ldots, T \tag{1}$$

We use $h_0 = 0$ in this work. Multiple such hidden layers can be stacked on top of each other, with x_t in equation 1 replaced with the activation at time t of the previous hidden layer, to obtain a 'deep' recurrent neural network. The output of the RNN at time t is computed using the state of the last hidden layer at t as

$$y_t = \theta(W_{yh} h_t^n + b_y) \tag{2}$$

where θ is a non-linear operation such as sigmoid or hyperbolic tangent for binary classification or softmax for multiclass classification, b_y is the bias term for the output layer and n is the number of hidden layers in the architecture. The output of the RNN at desired time steps can then be used to compute the error and the network weights updated based on the gradients computed using Back-propagation Through Time (BPTT). In simple RNNs, the function g is computed as a linear transformation of the input and previous hidden state, followed by an element wise non-linearity.

$$g(x_t, h_{t-1}) = \theta(W_{hx} x_t + W_{hh} h_{t-1} + b_h) \tag{3}$$

Such simple RNNs, however, suffer from the vanishing and exploding gradient problem [7]. To address this issue, a novel form of recurrent neural networks called the Long Short Term Memory (LSTM) networks were introduced in [7]. The key difference between simple RNNs and LSTMs is in the computation of g, which is done in the latter using a memory block. An LSTM memory

block consists of a memory cell c and three multiplicative gates which regulate the state of the cell - forget gate f, input gate i and output gate o. The memory cell encodes the knowledge of the inputs that have been observed up to that time step. The forget gate controls whether the old information should be retained or forgotten. The input gate regulates whether new information should be added to the cell state while the output gate controls which parts of the new cell state to output. The equations for the gates and cell updates at time t are as follows:

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \quad (4)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \quad (5)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \quad (6)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \phi(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \quad (7)$$

$$h_t = o_t \odot c_t \quad (8)$$

where $\odot$ is the element-wise multiplication operation, σ and ϕ are, respectively, the sigmoid and hyperbolic tangent functions, and h_t is the output of the memory block. Like simple RNNs, LSTM networks can be made deep by stacking memory blocks. The output layer of the LSTM network can then be computed using equation 2. We refer the reader to [7] for more technical details on LSTMs. The details of the architecture used in this work are described in section 5.2

3.6 Multimodal LSTM for liveliness classification

In the proposed approach, LSTMs are used to learn the discriminative visual and audio feature representations for liveliness. The estimates from audio and visual LSTMs are combined to estimate the overall liveliness of videos. For setup categories 'Person Walking/Standing', 'Person Sitting' and 'Content & Person' setup, both the modalities are used for liveliness prediction. For the remaining videos from 'Content' and 'Miscellaneous' categories, only the audio LSTM representation is used to determine the liveliness label. The details of the proposed approach are described below:

- **Visual-LSTM:** A multi-layer LSTM network is trained to learn the discriminative visual features for liveliness. The number of layers and the number of nodes in each layer in the LSTM network are determined based on a validation dataset. The input to the network at each time step t is a 512 dimensional visual feature extracted as described in 3.3.
- **Audio-LSTM:** The approach for training an audio LSTM is similar to that for training the visual LSTM. The only difference is that the visual features are replaced by the audio features as described in 3.4.
- **Multimodal-LSTM:** Once we learn the discriminative audio and visual LSTMs, the next step is to combine their predictions to determine the final liveliness. The visual and audio features from each 10-second clip are now forward-propagated through the visual-LSTM and audio-LSTM respectively. Once the features corresponding to all the time-steps of a clip have been forward-propagated, the liveliness prediction from each of these LSTM networks are obtained. If the setup corresponding to a clip requires combining audio and visual modality information, we assign the clip a positive liveliness label if any one of the visual-LSTM or Audio-LSTM network predicts the label of the clip as positive. Otherwise, the audio-LSTM label is used as the final label for the 10-second clip.

The proposed multimodal pipeline for liveliness prediction is called **LIVELINET** and will be referred as that from now on.

4. BASELINE DETAILS

In this section, we describe several baselines which do not use any deep neural network for feature extraction or classification. However, these methods have demonstrated state-of-the-art accuracy in many video/audio classification applications. We wanted to evaluate how good these "shallow" methods perform on the liveliness prediction task.

4.1 Visual Baseline

The visual baseline consists of training a SVM classifier on state-of-the-art trajectory features aggregated into local descriptors. Improved Dense Trajectories (IDT) [32] have been shown to achieve state of the art results on a variety of action recognition benchmark datasets. Visual feature points on the visual frames are densely sampled and tracked across subsequent frames to obtain dense trajectories. Once the IDTs are computed, VLAD (Vector of Locally Aggregated Descriptors) encoding [33] is used to obtain a compact representation of the video. We set the number of clusters for VLAD encoding at 30 and obtain a 11880-dimensional representation for each video. SVM classifier with RBF kernel is used for the classification. We compare this visual baseline against the proposed approach.

4.2 Audio Baselines

We compare LIVELINET with two different audio baselines; the first one uses bag of audio words and the second one uses Hidden Markov Models (HMM). The audio features are computed at a frame rate of 10 ms. The features are computed using the open source audio feature extraction software OpenSMILE [34]. Motivated by the findings in [35] and [36], where the authors show superior performance on various paralingustic challenges, our frame-level features consist of (a) loudness, defined as normalized intensity raised to a power of 0.3, (b) 12 Mel Frequency Cepstral Coefficients (MFCCs) along with the log energy ($MFCC_0$) and their first and second order delta values to capture the spectral variation, and (c) voicing related features such as the fundamental frequency (F0), voicing probability, harmonic noise ratio and zero crossing rate. (Intensity and fundamental frequency features have been found to be beneficial in liveliness classification in [4] also.) Authors in [36] refer to these frame-level features as Low Level Descriptors (LLD) and provide a set of 21 functionals based on quartile and percentile to generate chunk level features. We use all of these LLDs and the functionals for the audio feature extraction. For every one second audio signal (obtained using the same method as described in Section 3.4), these frame-level features are concatenated to form a $(44 * 100 = 4400)$ dimensional feature vector. The dimensionality of the chunk-level audio feature is further reduced to 400 by performing a PCA across all the chunks in the training data.

The audio features from all the one second audio signals in the training videos are clustered into 256 clusters. A nearest neighbour cluster centre is found for each of these audio features. We then create a 256-dimensional histogram for each clip based on these nearest neighbour assignments. This approach, known as the bag-of-words model is popular in computer vision and natural language

processing, and is beginning to be extended to the audio domain in the form of bag-of-audio-words (BoAW) (e.g., [37]). A SVM classifier with RBF kernel is trained on this BoAW representation.

As a second baseline, two 3-state HMMs, one each for the positive and the negative class, are trained using the sequence of audio features computed on these one second audio signals. Only left-to-right state transitions are permitted with a potential skip from the first state to the third state. Each state is modeled as 16-mixture Gaussian Mixture Model. The 44 frame-level LLD are the inputs to the HMM framework. The Scilearn implementation of HMM is used.

4.3 Multimodal baseline

For combining the audio and video modalities we employ a classifier stacking approach. Stacking involves learning an algorithm to combine the predictions of other classifiers. We first train two SVM classifiers on audio and video features separately. The features and kernels used here are the same as the individual audio and visual baselines described earlier. Subsequently, another SVM classifier (with RBF kernel) is trained on the predictions of the audio and video classifiers to make the final prediction. We compare this baseline against the proposed multimodal classifier.

5. EXPERIMENTAL RESULTS

In this section, we provide the details of the experimental results. First, we describe the StyleX dataset followed by the details of the proposed LSTM network architecture and setup classification results. Next, we provide the liveliness classification results using the proposed multimodal deep neural network method. Finally, we perform some preliminary quality analysis of the lively/not-lively videos.

5.1 Dataset

We use the StyleX dataset proposed in [6] for our experiments. StyleX comprises of 450 one-minute video snippets featuring 50 different instructors, 10 major topics in engineering and various accents of spoken English. Each video was annotated by multiple annotators for liveliness. The scores from all annotators (in the range $0 - 100$, where 0 implies least lively and 100 implies most lively) corresponding to a particular video were averaged to obtain the mean liveliness score. The bimodal distribution of the mean liveliness scores were analyzed to estimate the threshold for binary label assignment (lively and not-lively). All videos with liveliness score above the threshold were assigned to the positive class whereas the remaining videos were assigned to the negative class. At a threshold of 54, we have 52% videos in the negative class (Thus, a simple majority-class classifier would lead to 52% classification accuracy). Out of the 450 StyleX videos, we randomly choose 60% for training, 20% for validation and 20% for testing while ensuring a proportional representation of both the classes in each subset. Since the proposed method takes 10-second clips as input during training and testing, we further split each one-minute video into 10-second clips bookended by silence, with a 50% overlap across adjacent clips. Each of these 10-second clips are assigned the same label as the actual one-minute videos and are treated as independent training instances. Likewise, during test, the 10-second clips are extracted from one-minute videos. The label is predicted for each 10-second clip and the label of the one-minute video is determined based on the majority vote.

5.2 LSTM Architecture Details

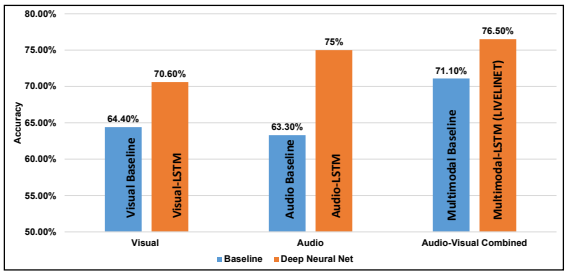


Figure 3: A comparison of results obtained from our proposed Multimodal-LSTM (LIVELINET) approach and the baselines.

The parameters of the proposed visual-LSTM and audio-LSTM were selected using the validation set. The learning rate was initialized to 10^{-4} and decayed after every epoch. Dropout rate of 0.2 was used for the activations of the last hidden layer. We tried nine different combinations for the number of hidden layers (1, 2, 3) and number of units in each layer (128, 256, 512), for both visual and audio modalities. Visual-LSTM with 2 layers and 256 hidden units and audio-LSTM with 2 layers and 256 hidden units led to the optimal performance on the validation set.

5.3 Setup Classification

In this section, we report the visual setup classification results obtained using the framework proposed in Section 3.2. As discussed in Section 5.1, the number of video clips used is 2700 for the training phase and 900 each for the validation and testing phase (all clips are approximately 10 seconds long). The network is trained with all the frames ($\sim$ 300K) extracted from the training video clips. At the time of testing, a label is predicted for each of the frame in a 10-second clip and their majority is taken as the label of the full clip. We evaluate 5-way classification accuracy of the video clips into different visual setups. Our proposed CNN architecture achieves a classification accuracy of 86.08% for this task. However, we notice that for the task of liveliness prediction, we only require the classification of video clips into two different classes - (a) clips requiring only audio modality, and (b) clips requiring both audio and video modality for liveliness prediction. For this task of binary classification ('Content or Miscellaneous' v/s 'Person Walking/Standing or Person Sitting or Content & Person'), our system achieves an accuracy of 93.74%. Based on the visual setup label of a clip, we use either both audio/visual or only audio modality for liveliness prediction.

5.4 Liveliness Classification

In this section, we present the performance of proposed multimodal deep neural network for liveliness prediction. Figure 3 depicts the results of our experiments. We obtain an accuracy of 70.6% with the Visual-LSTM, an absolute improvement of 6.2% over the visual baseline. The two audio baselines of HMM and BoAW methods lead to an accuracy of 60% and 63.3%, respectively. The Audio-LSTM setup leads to 75.0% accuracy, an increase of 11.7% over the best audio baseline. The proposed Multimodal-LSTM method (LIVELINET) achieves an accuracy of 76.5% compared to 71.1% obtained using the audio-visual baseline, an absolute improvement of 5.4% (relative improvement of 7.6%). We are also relatively 1.9% better than using only the audio-LSTM. The boost in accuracy when using both the modalities indicates that the information available from audio and visual modalities are complimentary and the proposed approach exploits it optimally.

5.5 Qualitative Analysis

We also perform qualitative analysis of the videos that are predicted lively/not-lively by LIVELINET. Our goal is to determine the general visual and audio patterns that make a video lively. These is the preliminary analysis of exemplar lively and exemplar non-lively lectures. We continue to perform a more systematic and in-depth qualitative analysis to understand two aspects: (a) patterns that the proposed classifier identifies as representative of lively and of not-lively, and (b) general audio-visual patterns that may have influenced the human labelers in assigning the 'lively or non-lively' label . One of the current directions for extending this work is to understand pedagogically-proven best practices of teaching and codify that knowledge in the form of features to be extracted and fed to the classifier. Some example frames from lively and not-lively videos as predicted by LIVELINET are shown in Figure 4. Some of our initial finding are: (a) Lecturers who alternate between making eye contact with the audience and looking at the content are perceived as more lively. (b) Similarly, voice modulations and moving around in the classroom (as opposed to sitting in place) and specific visual references (like pointing to written content) to synchronize with the spoken content seem to positively influence perceived liveliness.

6. CONCLUSION

We propose a novel method called LIVELINET that combines visual and audio information in a deep learning framework to predict liveliness in an educational video. First, we use a CNN architecture to determine the overall visual style of an educational video. Next, audio and visual LSTM deep neural networks are combined to estimate if a video is lively or not-lively. We performed experiments on the StyleX dataset and demonstrated significant improvement compared to the state-of-the-art methods. Future directions include incorporating text-based features for a content-based liveliness scoring. We also note that LIVELINET is going to be part of our e-learning platform TutorSpace.

References

[1] P. J Guo and K. Reinecke. Demographic differences in how students navigate through moocs. In *LAS*. ACM, 2014.

[2] J. Kim, P. J Guo, D. T Seaton, P. Mitros, K. Z Gajos, and R. C Miller. Understanding in-video dropouts and interaction peaks inonline lecture videos. In *LAS*. ACM, 2014.

[3] S S. Krishnan and R. K Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. *Networking, IEEE/ACM Transactions on*, 2013.

[4] S. Marioryad, A. Kannan, D. Hakkani-Tur, and E. Shriberg. Automatic characterization of speaking styles in educational videos. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014.

[5] Rebecca Hincks. Measures and perceptions of liveliness in student oral presentation speech: A proposal for an automatic feedback mechanism. *System*, 33(4):575–591, 2005.

[6] H. Arsikere, S. Patil, R. Kumar, K. Shrivastava, and O. Deshmukh. Stylex: A corpus of educational videos for research on speaking styles and their impact on engagement and learning. In *INTERSPEECH*, 2015.

[7] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[8] R. J Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1989.

[9] A. Graves and J. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *NIPS*, 2009.

[10] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. In *Human Behavior Understanding*. Springer, 2011.

[11] X. Chen and Lawrence Z. C. Mind's eye: A recurrent visual representation for image caption generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[12] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[13] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. *stat*, 2015.

[14] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*. IEEE, 2013.

[15] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *ICML*, 2014.

[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[17] A. Krizhevsky, I. Sutskever, and G. E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[18] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, and Xiangyang Xue. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference*, pages 461–470. ACM, 2015.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2015.

[20] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from RGB-D images for object detection and segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, 2015.

[22] Seunghoon Hong, Tackgeun You, Suha Kwak, and Bohyung Han. Online tracking by learning discriminative saliency map with convolutional neural network. *CoRR*, 2015.

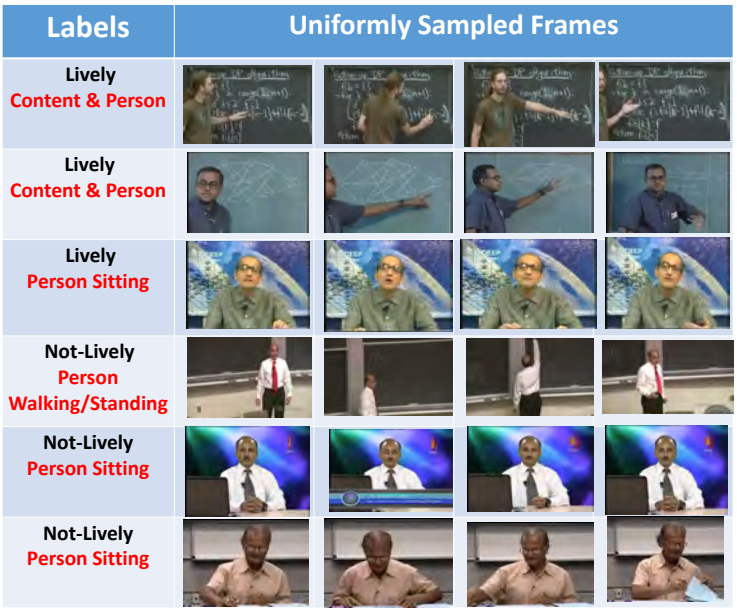


Figure 4: Some example frames from videos predicted as lively and not-lively by our proposed method LIVELINET. The setup labels predicted by the proposed Setup-CNN approach are also shown.

[23] Tian Gan, Yongkang Wong, Bappaditya Mandal, Vijay Chandrasekhar, and Mohan S. Kankanhalli. Multi-sensor self-quantification of presentations. In *Proceedings of the 23rd ACM International Conference on Multimedia*, 2015.

[24] A. T. Nguyen, W. Chen, and M. Rauterberg. Online feedback system for public speakers. In *E-Learning, E-Management and E-Services (IS3e), 2012 IEEE Symposium on*, 2012.

[25] Vanessa Echeverría, Allan Avendaño, Katherine Chiluiza, Aníbal Vásquez, and Xavier Ochoa. Presentation skills estimation based on video and kinect data analysis. In *Proceedings of the 2014 ACM Workshop on Multimodal Learning Analytics Workshop and Grand Challenge*, MLA '14, 2014.

[26] Lei Chen, Gary Feng, Jilliam Joe, Chee Wee Leong, Christopher Kitchen, and Chong Min Lee. Towards automated assessment of public speaking skills using multimodal cues. In *Proceedings of the 16th International Conference on Multimodal Interaction*, 2014.

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[28] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[29] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[30] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014.

[31] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. In *Proceedings of the 29th DAGM Conference on Pattern Recognition*, 2007.

[32] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*. IEEE, 2013.

[33] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*. IEEE, 2010.

[34] Florian Eyben, Martin Wöllmer, and Björn Schuller. Opensmile: The munich versatile and fast open-source audio feature extractor. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 1459–1462, New York, NY, USA, 2010. ACM.

[35] Björn Schuller, Stefan Steidl, Anton Batliner, Felix Burkhardt, Laurence Devillers, Christian A Müller, and Shrikanth S Narayanan. The interspeech 2010 paralinguistic challenge. 2010.

[36] Björn Schuller, Stefan Steidl, Anton Batliner, Alessandro Vinciarelli, Klaus Scherer, Fabien Ringeval, Mohamed Chetouani, Felix Weninger, Florian Eyben, Erik Marchi, et al. The interspeech 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism. 2013.

[37] Stephanie Pancoast and Murat Akbacak. Bag-of-audio-words approach for multimedia event classification.

Semantic Features of Math Problems: Relationships to Student Learning and Engagement

Stefan Slater
Ryan Baker
Jaclyn Ocumpaugh
Teachers College Columbia University
525 W. 120th St
New York, NY 10027
{slater.research, ryanshaunbaker, jlocumpaugh}@gmail.com

Paul Inventado
Peter Scupelli
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
{pinventado, scupelli}@cmu.edu

Neil Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609
nth@wpi.edu

ABSTRACT
The creation of crowd-sourced content in learning systems is a powerful method for adapting learning systems to the needs of a range of teachers in a range of domains, but the quality of this content can vary. This study explores linguistic differences in teacher-created problem content in ASSISTments using a combination of discovery with models and correlation mining. Specifically, we find correlations between semantic features of mathematics problems and indicators of learning and engagement, suggesting promising areas for future work on problem design. We also discuss limitations of semantic tagging tools within mathematics domains and ways of addressing these limitations.

Keywords
Text mining, semantic analysis, problem features, engagement, learning, correlation mining, mathematics corpora

1. INTRODUCTION
As content is developed at scale for online learning systems, particularly systems that leverage content developed by large numbers of authors, it becomes important to distinguish between problems which are well-written and conducive to learning and those which are poorly worded or otherwise difficult to understand. Crowd-sourced content, where content is authored by a broader community [21], is a powerful and scalable method of content creation, which can be used to quickly develop and deploy new content and curricula ([46], [17]).

For this reason, it is critical that an equally scalable method of analyzing problem quality be developed, to prevent learning platforms that leverage crowd-sourced content from becoming dominated by ineffective content. In other platforms such as Wikipedia the quality of crowd-sourced materials is improved through substantial coordination between contributors [20]. However, there is relatively little work evaluating crowd-sourced learning content at scale. In contrast with more traditional educational measurement (from tests), where determining items' ability to discriminate student knowledge is a standard part of item analysis [11], there has been less attention to this problem for online learning systems. While some researchers have attempted to determine which hints are more effective [18], or which problems are associated with more learning [14], these efforts have focused on what, but not why, particular system features can impact student, limiting their degree of general use. A more theoretical approach was taken by [49] where a design space of over 70 features characterizing Cognitive Tutor lessons was distilled and correlated with an automated gaming the system detector. However, this work identified the characteristics of tutor lessons using hand-coding, a method that is infeasible for larger datasets, and was limited to the relatively narrow space of problems designed by professional educational developers.

An alternative method for the analysis of the design of content in large-scale educational systems is text mining. There is a considerable amount of small-scale research on linguistic features that impact reading in mathematical contexts [47], but as [16] point out, many of the traditional readability indices used to study language at scale are limited in the features they consider. As a result, many early studies did not find a relationship between readability and performance in mathematics word problems [48].

As more advanced linguistic tools have become available, large-scale investigations of mathematics language have become more fruitful. For example, [44] have used LIWC [37] and CohMetrix [15] to study the effects of linguistic properties of mathematics problems ([44], [45]). [45] found that third-person singular pronouns (e.g., he, she) are significantly associated with correct answers and fewer hint requests in Cognitive Tutor problems. They found positive correlations between the use of work-related terms and learning, and negative correlations between the use of terms related to social constructs and learning. These findings highlight the potential value of linguistic features for better understanding learning, as well as the need to explore a wider range of semantic categories in a broader range of mathematics content areas.

In this paper, we use a discovery with models approach, generating prediction labels from automated detectors of student learning and engagement that were developed for the ASSISTments online learning system ([2], [32]). We build on [46]'s approach of using text mining software and text elements, such as HTML tags and Unicode characters, to distill features

EDM '16, June 29–July 2, 2016, Raleigh, NC, USA.

from a corpus of mathematics problems. We then use correlation mining approaches to identify links between these features and our labels of student engagement and learning as a means for determining which combinations of linguistic features are associated with particularly effective problems.

1.1 ASSISTments

The current study uses data collected from the ASSISTments system. ASSISTments is an online intelligent tutoring system used by over 50,000 students annually for middle-school mathematics. It provides both formative and summative *assessment* as well as extensive student support (*assistment*) and detailed teacher reports. It also facilitates research using randomized controlled trials (RCTs) that allow researchers to conduct studies without interfering with instructional time [17].

Within the system, students are assigned problem sets that may vary on several dimensions. Problem sets can be differentiated in terms of how problems are assigned: (a) In *Complete All* problem sets, problem order may be randomized; students must correctly answer all of the questions assigned and cannot advance to the next problem unless they have answered correctly. (b) In *If-Then-Else* problem sets, students must correctly answer a specified percentage of questions correctly (default is 50%) in order to pass, or *else* they may be given additional problems. (c) Finally, in *Skill Builder* problem sets, students must get 3 consecutive correct answers in order to pass, thus allowing students who show mastery to move on quickly to new assignments while providing struggling students with extended practice.

The purpose of the current study is to evaluate the semantic properties and HTML metadata (which may carry semantic meaning) of problems authored in ASSISTments. Many have been vetted by the ASSISTments expert team, but others (76% as of 2014) were created by teachers themselves [17]. ASSISTments provides scripted templates, which allow teachers to customize problem sets for specific topics. Therefore, finding ways to identify meaningful differences in teachers' problem design is an important area of research.

2. DATA & METHODS

In this paper, we analyze 179,908 problems within the ASSISTments system, most developed by teachers. We study these problems using the features of the problems themselves, in combination with data from the log files of 22,225 students who used ASSISTments during the 2012-13 school year. We applied models from previous research on engagement and learning to these students' log files in order to determine how these constructs are associated with features of the design of the problems, developed through linguistic analysis and other data about the problems. In doing this, we excluded from consideration features that had been previously used within the learning and engagement models described below, to prevent overfitting.

2.1 Learning & Engagement Measures

Learning and engagement were assessed automatically, using detectors or models of these constructs.

2.1.1 Student Learning

Student learning was assessed by fitting the moment-by-moment learning model to the data [2]. The moment-by-moment learning model (MBMLM) attempts to infer the specific effect of each learning opportunity on a student's overall mastery. We used [2]'s look-ahead-two probabilistic approach, which assumes that learning can occur at multiple points along a student's trajectory of learning a skill, rather than [43]'s approach which assumes a single moment of learning. We also choose this formulation because it explicitly analyzes future performance, allowing us to focus on cases where students perform better than expected after encountering a particular problem. Using the MBMLM allows us to isolate the average learning associated with specific problems within the data and compare these averages to other problems that either lack or have particular features of interest.

2.1.2 Automated Detectors of Engagement

Detectors of student engagement were developed using data from *in situ* classroom observations, conducted by experts certified in the Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP 2.0). The protocol is enforced by HART, an Android application designed specifically for the BROMP and freely available for non-commercial research [33], which enforces the protocol while facilitating data collection.

Upon completion of the observations, data mining techniques were then employed to provide models of each construct that were cross-validated at the student level. In this paper, affective models developed for three different populations of students were applied, matching urban, suburban, and rural models to student data based on the location of their schools, in order to ensure population validity [32]. A detailed description of the features and algorithms used in these detectors is given in [32] and [34].

2.1.3 Applying Across-Student Measures of Learning & Engagement to Individual Problems

In this paper, both the MBML model and the engagement models were used as indicators of problem effectiveness. This section describes how these models were aggregated across the 179,908 problems and 22,225 students in this study. The formulation of the MBMLM in [2] is calculated once for each problem, at the time of the first attempt, and there is only one estimate per problem. Therefore, MBML was estimated for each student based on the sequence in which the problem was seen. Problem-level measures were then produced by averaging the MBML values across all students who saw a given problem.

The affective models were applied by segmenting the data at 20-second intervals (matching the original approach used to develop the detectors), and then applying each model to each segment. Confidence values for each detector was averaged twice at the problem level: first for each student (in order to avoid biasing the estimates in favor of the affect experienced by students who spent longer working the problem), then across all students who had seen that problem. This resulted in five measures per problem (average boredom, confusion, engaged concentration, frustration, and gaming), which we used, along with MBMLM outcomes, as our dependent variables.

2.2 Feature Engineering

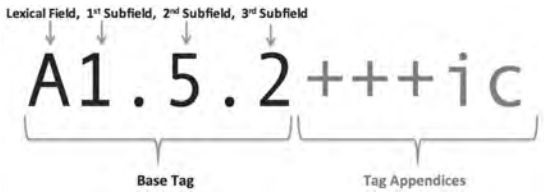
A number of different design features may influence student learning and engagement. In this paper, we explore features of both the problem text and its meta-text. Specifically, we look at word counts, lexical category features generated by a semantic tagger, and features generated from the metadata connected to the problem, which provides us with a separate source of semantic data (e.g., the use of mathematical notation which would not be captured by a semantic tagger) as well as with information about its use of tables, images, formatting, bolded or emphasized text.

2.2.1 Wmatrix Semantic Tags

The semantic content of ASSISTments problems was analyzed with Wmatrix [39], a corpus analysis and comparison tool that

parses text at a word and multi-word level. As of 2004, this included 42,300 single word entries and over 18,400 multi-word expressions [38]. Wmatrix has been used in a number of analyses, including work to tag and identify lexical patterns in ontology learning [13] and work to study how students self-explain when learning science content [12]. Its semantic tagger uses a semi-hierarchical structure where all known words and multi-word units are classified into one of 21 lexical fields, represented with letters by its tagging system. These lexical fields may (or may not) be further subdivided in up to three different levels, which are represented in what we will refer to as the base tag.

Figure 1. WMatrix tagging system.



Within the lexical tag, we will refer to the lexical field (alphabetical) and the 1st, 2nd, and 3rd order subfields (numeric) as the *base tag*. Additional information about antonyms (*black* vs. *white*), comparatives (*better, worse, more confusing*, etc.), superlatives (*best, worst, most confusing*, etc.), gender (*masculine, feminine*, and *neuter*), and anaphoric status (i.e., contextual reference), may or may not be *appended* to a base tag. Wmatrix documents 234 distinct base tags, and represents a large number of additional possible labels through appendices

In the ASSISTments data, 442 distinct Wmatrix tags (base + appendices) were identified. These tags were most likely to fall under 7 lexical fields: General & Abstract Terms (A), Numbers & Measurement (N), Social Actions, States, & Processes (S), Psychological Actions, States, & Processes (X), Names & Grammatical Words (Z), Money & Commerce in Industry (I), and Time (T).

2.2.2 Accommodating Known Wmatrix Limitations

Although Wmatrix has been evaluated for its effectiveness in a range of genres, domains, and historical periods [38], semantic taggers can have a number of limitations when applied to highly specialized domains ([28], [24]; [36]; [30]; [27]). For example, research has shown that words which contain more than one unit of meaning create challenges for taggers that apply only one label per word [41]. As a result, semantic taggers which work specifically with scientific language have become an area of research interest ([1], [10]), but the language of mathematics has not yet been as prominent.

As such, features generated by Wmatrix must be carefully checked within this data set and may need to be supplemented by domain-specific tags. For example, we found several Wmatrix tags that erroneously tagged high-frequency items that appeared in ASSISTment's instructions to students, including problems that instructed students to *enter* fractions in a specific format in order to receive credit or which told students that they had 3 attempts *left.* Wmatrix treated many of these words (e.g., *enter* and *left)* as an indication of physical movement (M1, as in *entering a building* or *turning left*). A few erroneous tags also appeared to result from the development of Wmatrix as a tool for British English. For instance, ASSISTments users, who are primarily American English speakers, wrote a number of problems involving a person named *Randy*, whose name was automatically (and erroneously) tagged as involving sexual content.

To mitigate this issue, significant correlations were carefully inspected individually. This approach has been found to be useful in previous studies where semantic taggers were applied to new domains [12]. While the large size of the ASSISTments corpus limits our ability to address this problem completely, thorough efforts were made to examine and understand relationships discovered through the use of Wmatrix. In instances where Wmatrix applied a tag involving the wrong sense of a word for the context in which it was used, we have specifically noted this difference and what sense of a word or words the tag is capturing within ASSISTments.

2.2.3 Math Symbols and Other Textual Metadata

In addition to generating features with Wmatrix, we also generated features based on the metadata of each problem. We were primarily concerned with identifying Unicode characters that are semantically meaningful in mathematics contexts. In the ASSISTments corpus, we labeled 68 symbols, such as those for integrals, mean, standard deviation, and exponents. These domain-specific symbols present unique challenges to the teaching and learning of mathematics [40], but are not detected by most lexical analysis tools, which have not generally been developed for mathematics domains. In addition, we identified 14 HTML tags that were used to format ASSISTments problems, including tags used for boldface, italics, paragraph structure, and images. Because many of these functions can also alter the semantics of a problem, we also generated features that reflect these uses of HTML in problem metadata. These features were generated by counting the number of times that each HTML code was used in a problem, in parallel to the application of the Wmatrix tags discussed in previous sections.

3. RESULTS

To explore the relationship between these problem features and the BROMP-trained measures of engagement and learning, we correlated each problem feature to each predicted variable. We selected Spearman's ρ as our correlation coefficient because of its increased robustness when correlating non-normal data as compared to other parametric coefficients such as Pearson's R [50]. Additionally, with such a high number of comparisons being conducted it was necessary to adjust our significance criterion to account for the possibility of tests being incorrectly identified as significant. The Benjamini and Hochberg post-hoc procedure [4] was used to control for these false discoveries. A table of results by dependent variable is presented in Table 1, which also provides the average confidence level for each detector as a baseline measure for this data.

Table 1. N of significant features by outcome measure.

Outcome Measure	Avg Conf.	Total Sig	Sig w/ $\|\rho\| >$ 0.05	Sig w/ $\|\rho\| >$ 0.10
Bored	0.16	118	16	0
Engaged Concentration	0.46	251	62	14
Confusion	0.03	285	60	5
Frustration	0.04	216	36	7
Gaming the System	0.02	257	43	5

Of the possible 2730 correlations, 1127 (41.3%) were statistically significant after controlling for multiple comparisons using Benjamini & Hochberg's post-hoc control. More features were

significantly correlated with confusion than any other outcome measure, but large numbers of features were also correlated with gaming the system, engaged concentration, frustration and MBML. Boredom was correlated with fewer features, overall, than either of the other outcome measures. These broad findings suggest the potential for finding semantic features that may help to provide templates for improving the design of word problems.

3.1 Features associated with all outcome measures

In the following sections, we examine the relationships between our features and the individual outcome measures, but in order to provide a broad summary of which types of features had the largest effects, the absolute value of Spearman ρ was averaged across all six outcome measures for each feature in this study. Among the 64 features that were signifcantly correlated with all six outcomes, the 10 with the highest ρ average (shown in Table 2) were drawn from 5 lexical fields: Grammatical Bin (Z), General Terms (A), Time (T), Speech Acts (Q), and Numbers & Measurement (N). One HTML tag (<p>, paragraph) was also significant.

Table 2. 10 largest correlated features by average sig. |ρ|

Tag	Avg |ρ|	MBM Learning	Boredom	Concentration	Confusion	Frustration	Gaming
Z5	0.116	0.193	0.086	-0.165	0.084	0.105	0.060
Z5mwu	0.104	0.114	0.034	-0.040	0.135	0.162	0.140
A12-	0.101	0.114	-0.027	0.030	0.086	0.153	0.198
T3-	0.091	0.084	-0.034	0.055	0.074	0.144	0.153
Q2.2	0.080	0.043	0.083	-0.162	0.068	0.071	0.051
T1.1.2	0.076	0.076	-0.051	0.031	0.067	0.116	0.116
<p>	0.071	0.149	0.054	-0.127	0.015	0.064	-0.015
N1	0.069	0.061	0.076	-0.077	0.082	0.080	0.035
A5.4+	0.066	-0.028	0.059	-0.130	0.074	0.038	-0.069
Z6	0.056	0.108	0.020	-0.034	-0.077	-0.032	0.071

Spearman's ρ is also shown for individual outcome measures, allowing us to examine the effects of these features in greater detail. Table 2 shows that WMatrix's Speech Acts tag (Q2.2, e.g., *answer*, *account*, or *speak out*) is correlated with small increases in learning, but is also positively correlated with increased boredom and gaming and decreased concentration. The Wmatrix features described as *Grammatical Bin* (words such as *as, but, in order to*) are also correlated with increased learning, boredom, and gaming. Correspondingly, they are also negatively associated with engaged concentration, illustrating the complicated interactions at play in this data and the importance of considering multiple outcomes when exploring design effects.

4. Results by Outcome Measure

While some interactions are complicated, we also see many features correlate in logical patterns. For example, features that are positively associated with boredom are often also negatively associated with engaged concentration, and vice-versa. Likewise, features associated with confusion are also associated with frustration. The remainder of this section discusses these patterns in greater detail, pairing outcome measures that are conceptually related (e.g., boredom and engaged concentration as well as MBML and gaming the system, which have shown to be inversely related in the past). Specifically, we will examine the ten features that are most negatively associated and the ten that are most positively associated with each outcome measure, discussing commonalities across outcome measures.

4.1.1 Learning & Gaming the System

The Spearman ρ values for the top ten features range from -0.078 to 0.233 for MBML and from -.095 to 0.198 for gaming the system. Table 3 presents these results, highlighting features that correlate with both outcome measures.

Table 3. Features most strongly associated with MBML and gaming the system

LEARNING			GAMING		
TAG	**Semantic Description**	**ρ**	**TAG**	**Semantic Description**	**ρ**
A5.2+	True/False	-0.078	N5+	Quantities	-0.095
S9	Religion & the supernatural	-0.075	A10+	Open/Closed; Hiding/Hidden; Findin	-0.092
A11.1+++	Important/Significant	-0.066	X2.1	Thought/belief	-0.084
A6.1+	Similar/Different	-0.062	A2.1+	Modify, Change	-0.082
G2.2+	General Ethics	-0.059	S5+	Groups and affiliation	-0.074
N3.2+++	Measurement: Size	-0.059	N5.2+	Exceeding; waste	-0.070
A3-	Being	-0.058	A5.4+	Authenticity	-0.069
Z8mwu	Pronouns etc.	-0.054	T1	TIME GENERAL	-0.069
N1mwu	Numbers	-0.051	N5	Quantities	-0.067
X5.2+	Interest/boredom/excited/ennergetic	-0.049	T2+	Time: Beginning and ending	-0.067
A12-	Easy/Difficult	0.114	A7+mwu	Definite (+modals)	0.086
Z5mwu	Grammatical bin	0.114	X2.4mwu	Investigate/examine/test/search	0.087
Z99	Unmatched	0.114	N3.8+	Measurement: Speed	0.093
N3.3---	Measurement: Distance	0.115	Z8	Pronouns etc.	0.093
X2.2+	Knowledge	0.121	A12+++	Easy/Difficult	0.098
M7	Places (geographical & conceptual)	0.130	T1.1.2	Time: General: Present; Simultaneous	0.116
N3.8+	Measurement: Speed	0.142	X8+	Trying	0.140
<p>	HTML paragraph	0.149	Z5mwu	Grammatical bin	0.140
Z5	Grammatical bin	0.193	T3-	Time: Old, new and young; age	0.153
M1	Moving, coming, & going	0.223	A12-	Easy/Difficult	0.198

Although gaming is an infrequent behavior, previous research has shown that it is linked to poorer learning ([7], [34]). Therefore the findings in Table 3 are somewhat surprising. We should expect gaming's infrequency to limit overlap between the two categories, and expect them to show inverse relationships when present. Instead, A12- (words related to *difficulty*), Z5mwu (multiword grammatical units like *as far as* or *for example*), and N3.8+ (words related to *higher speeds*), are all associated with increased MBML **and** increased gaming behaviors. Likewise, semantically similar categories like N1mwu (multiword *numbers*) and N5+ (*large quantities*) are associated with lowered MBML **and** lowered rates of gaming behaviors.

These anomalies might be due to the existence of problems that support learning but can be gamed relatively easily, or might suggest that particularly challenging problems lead to learning but also inspire gaming behavior. For example, A5.2+ (words associated with *true*) demonstrates the lowest correlation with learning, a result that is consistent with literature on the ineffectiveness of true/false questions [42]. Likewise Z8mwu (multiword pronouns, e.g., *anything at all*) is correlated with lower MBML, while Z8 (single word pronouns, e.g., *it, my,* and *you*) is correlated with increased gaming. These findings align with research showing that pronouns can be difficult to process cognitively (taxing working memory), as they require readers to infer their antecedents (the words that give them their meaning) from context ([25], [8], [22], [6]). This suggests that pronouns could inhibit learning by drawing mental resources away from mathematics task, perhaps inspiring some students to try to succeed with minimal cognitive effort.

These findings highlight important considerations for researchers working to improve learning systems, including the need to consider multiple measures. For example, [44] found that pronouns are associated with correct answers and lowered hint use. It is highly likely that pronouns can have beneficial impacts on learning, particularly through [44]'s hypothesized mechanism of increased cohesiveness. However, if pronoun use in ASSISTments and Cognitive Tutor is comparable, our results suggest that some correct answers could have been achieved by guessing rather than by learning.

Furthermore, if students are more tempted to game the system when presented with challenging problems, even though these are exactly the sort of problems needed to improve learning, then further research should explore whether or not these findings reflect two distinct different groups of students. It may be that some students need additional cognitive scaffolding or a motivational intervention in order to complete these problems without gaming, allowing them to learn as well as other students who are working through the curriculum in a more appropriate way. However, research has also shown that in some cases high achieving students also game the system, and the independent application of these models could be picking up on that trend, where students guess something that they actually know, but then correct this behavior in subsequent problems, which could cause the MBML model to perceive learning.

4.1.2 Confusion & Frustration

Confusion and frustration show considerable overlap, in line with prior theory on the relationship between these constructs ([9], [26]). As Table 4 shows, half (10) of the semantic features most strongly associated with one are also strongly associated with the other, including N6mwu (*frequency of occurrence*) which is negatively associated with both confusion and frustration. This corresponds with [44]'s findings that clear demarcations of time in mathematics problems can improve student outcomes.

Table 4. Features most strongly associated with confusion and frustration

CONFUSION			FRUSTRATION		
TAG	**Semantic Description**	**ρ**	**TAG**	**Semantic Description**	**ρ**
X2.1	Thought/belief	-0.149	X2.1	Thought/belief	-0.110
Z6	Negative	-0.101	N5+	Quantities	-0.070
N3.4	Measurement: Volume	-0.097	A11.1+++	Important/Significant	-0.063
N3.3---	Measurement: Distance	-0.079	N3.4	Measurement: Volume	-0.061
N6mwu	Frequency of occurance	-0.079	A2.2	Cause, Connected	-0.056
A2.2	Cause, Connected	-0.077	N6mwu	Frequency of occurance	-0.052
A1.5.1	Using	-0.076	X4.2	Means, method	-0.051
N5+	Quantities	-0.070	T2++	Time: Beginning and ending	-0.050
I1.3	Money: price	-0.068	A2.1+mwu	Modify, Change	-0.049
O4.1	General Appearance/Phys'l Proper	-0.066	<font>	HTML font adjustment	-0.049
Q1.2mwu	Paper documents & writing	0.081	I3.1	Work & Employment: generally	0.089
N1	Numbers	0.082	X2.4mwu	Investigate/examine/test/search	0.092
I3.2	Work & Employment: professiona	0.083		HTML span (grouping of items in or	0.092
Z5	Grammatical bin	0.084	N6+	Frequency of occurance	0.093
A12-	Easy/Difficult	0.086	Z5	Grammatical bin	0.105
	HTML italics	0.087	T1.1.2	Time: General: Present; simultaneous	0.116
I3.1	Work & Employment: generally	0.094	T3-	Time: Old, new and young; age	0.144
S6+	Obligation and necessity	0.105	X8+	Trying	0.148
X8+	Trying	0.115	A12-	Easy/Difficult	0.153
Z5mwu	Grammatical bin	0.135	Z5mwu	Grammatical bin	0.162

Notable semantic features within this pairing include Z5 and Z5mwu. Both capture what are known as grammatical bin, which includes prepositions (*of, to, after, amid*), conjunctions (*and, or, but*), certain adverbs (e.g., *as, so, which, than, when*), the infinitival maker (*to* + verb), determiners (e.g., *a* and *the*) and certain auxiliary verbs (e.g., *do*). Previous research has suggested that the highly specific style of scientific language increases the use of these parts of speech, especially in the sort of definitional contexts that we might find in many learning contexts [3]. [29], for example, notes that students sometimes struggle with prepositions. In fact, this pattern is sometimes referred to as the *stylistic barrier hypothesis* [31], which suggests that differences between the language students use at home and the language used in the classroom may interfere with the learning process.

HTML features that that correlate with confusion and frustration match findings in the literature. For example, [35] suggest that italics are difficult to read, and our findings show that they are correlated with higher confusion. Changes in font size, however, are associated with lower frustration; it is possible that teachers are using changes in font size to clarify visual hierarchy and problem meaning.

Features associated with concreteness (N3.4, N3.3, A2.2, A1.5.1, N5+, I1.3, O4.1, T2++) correlate with lowered confusion and frustration, matching the literature on the *concreteness effect,* which shows that concrete words are not only processed faster than abstract words in many experimentally controlled studies [23], the two may operate in separate neurological pathways ([19], [5]). These findings are hypothesized to be an artifact of the word-to-word mapping system the brain uses to process language, where concrete words may have stronger ties to more basic concepts. Interestingly, [23] have found evidence for similar pathways for emotion words, which are acquired early and considered quite basic to the human experience. While several of the Wmatrix categories that might correspond with [23]'s account of emotion words do not appear in this list (E3, E4, X4.1), X2.1, described as *thoughts/beliefs*, has the strongest negative associations with both frustration and confusion.

Other features which correlate with increased confusion and frustration may reflect the sort of meta-instructions teachers use to support students working with complex mathematical problems. Consider, for example, the tags in the following examples:

(1) *You*_Z8mf *must*_**S6+** *show*_A10+ *your*_Z8 *work*_**I3.1**.
(2) *You*_Z8mf *have*_A9+ *three*_N1 *attempts*_**X8+**
(3) *Often*_**N6+** *it*_Z8 *helps*_S8+ *to*_**Z5** *write*_Q1.2[i1.2.1 *down*_Q1.2 [i1.2.2 *your*_Z8 *work*_**I3.1**.
(4) *Keep*_A9+ *trying*_**X8+**
(5) *Do*_**X8+**[i1.3.1 *your*_**X8+**[i1.3.2 *best*_**X8+**[i1.3.3
(6) *Do*_A1.1.1 *the*_**Z5** *difficult*_**A12-** *problems*_**A12-** *first*_N4

Several of these tags (as given in bold, above: I3.1 *work;* S6+ *must;* Z5 *to, the;* X8+ *attempts, trying*; A12- *difficult;* N6+ *often*) are correlated with increased confusion or frustration. This finding may reflect a preemptive scaffolding practice (e.g., teachers provide these additional instructions when students are working on problem types that they have struggled with in the past). However, it is important to rule out other possibilities. For instance, such additional instructions could distract or annoy the students. More seriously, it could also have priming effects.

4.1.3 Engaged Concentration & Boredom

Like confusion and frustration, we see considerable overlap in the features correlated with engaged concentration and boredom. However, unlike confusion and frustration, these two outcome measures are negatively associated with one another. Six of the features most negatively associated with concentration (N5-, N3.6, Z5, Q2.2, A4.1, and A5.4+) are among those most positively associated with boredom. Likewise, four of those most positively associated with concentration (A2.1+mwu, A6.1+++, T3, and A5.2+) are negatively associated with boredom.

Table 5. Features most strongly associated engaged concentration and boredom

ENGAGED CONCENTRATION			BOREDOM		
TAG	SEMANTIC DESCRIPTION	ρ	TAG	SEMANTIC DESCRIPTION	ρ
N5-	Quantities	-0.182	T1.1.2	Time: General: Present; Simultan'us	-0.051
N3.6	Measurement: Area	-0.178	A5.2+	True/False	-0.041
Z5	Grammatical bin	-0.165	X2	Mental actions & processes	-0.041
Q2.2	Speech Acts	-0.162	A2.1+mwu	Modify, Change	-0.034
A4.1	Generally/kinds/ groups/examples	-0.161	M6mwu	Location & Direction	-0.034
	HTML italics	-0.144	T3-	Time: Old, new and young; age	-0.034
A6.3+	Variety	-0.143	A8	Seem/Appear	-0.030
A5.4+	Authenticity	-0.130	T2++mwu	Time: Beginning and ending	-0.028
<p>	HTML paragraph	-0.127	A11.1+++	Important/Significant	-0.027
Z7	If	-0.116	A6.1+++	Similar/Different	-0.027
A4.2+	Particular/general; details	0.068	A5.4+	Authenticity	0.059
N3.5	Measurement: Weight	0.069	Z8c	Pronouns etc.	0.061
N3.1	Measurement: General	0.074	A6.3+	Variety	0.063
S5+c	Groups and affiliation	0.074	N1	Numbers	0.076
A2.1+mwu	Modify, Change	0.075	S6+	Obligation and necessity	0.076
A6.1+++	Similar/Different	0.077	N5-	Quantities	0.078
T3	Time: Old, new and young; age	0.082	Q2.2	Speech Acts	0.083
A2.1+	Modify, Change	0.083	A4.1	Generally/kinds/ groups/examples	0.085
Y1	Science/technology general	0.112	Z5	Grammatical bin	0.086
A5.2+	True/False	0.115	N3.6	Measurement: Area	0.093

Interestingly, X2.1 (*thoughts/beliefs*) is not as closely related to boredom and engagement as it was to confusion and frustration, but two other features typically associated with language about humans show desirable associations with these two outcome measures. For instance S5+c (*groups & affiliation*) is associated with increased engaged concentration, while X2 (*mental actions/processes*) is associated with lowered boredom. Likewise A8, which tags words related to *seem* or *appear* (both mental processes typically ascribed to human subjects), also leads to lowered boredom.

These semantic features, along with several others that correlate with lowered boredom (T2++mwu *time demarcations* and M6mwu *location/direction*) may also be indicators that problems with greater narrativity improve student engagement. However, we must still be cautious about interpreting lower boredom as a desirable effect in and of itself, since A5.2+ (words associated with *true*) is also associated with lower boredom. This type of item is unlikely to bore students, since they can answer and pass it quickly. However, readers may recall that this feature is also correlated with lower learning, as one might expect based on previous research on True/False questions [42].

5. DISCUSSION AND CONCLUSIONS

Our analyses of the ASSISTments corpus complements previous research on the relationship between learning and the language of mathematics problems, but extends this line of inquiry by including educationally relevant behaviors and affective states as part of the learning outcomes measured. As discussed, a number of linguistic features (e.g., *pronoun*s, *mental states, time*, and *concreteness*) have been found to be significant in previous work. However, we were also able to examine the degree to which these relationships reflect expectations about how behavior, affect, and learning are related.

For instance, some of the same features which were correlated with learning were also correlated with student frustration and gaming the system. While it might be hypothesized that frustrated students would be more likely to game the system, there is also evidence from within ASSISTments that frustration can be important for learning [26]. The MBML model used here is a look-ahead algorithm, which may optimize the opportunity to identify the problems that trigger learning even when learning process is causing student frustration. However, it's also possible that these problems are triggering strong but distinct reactions in different students (e.g., students who persist vs. students who game the system when they become frustrated). Future work will hopefully shed more light on this unusual relationship.

Overall, these results point to a number of promising avenues for further research within the ASSISTments system. One key future approach will be to conduct RCTs of the features identified in this study, re-designing problems to eliminate problematic features or incorporate positive features, in order to determine whether our findings can drive enhanced design. At the same time, it will be important to explore some of the interactions that may exist between different combinations of linguistic features, or between linguistic features and other behaviors or actions within the tutor.

We also found several unusual patterns in our data, such as some features being associated with increases in both learning and with gaming the system. We believe this may be due to our dataset containing two different populations of students – those who are persistent in the face of challenging and difficult problems and those who are frustrated by these problems and attempt to game the system to avoid working through them. We hope to understand this relationship in greater detail through RCTs (as discussed below). Ultimately, we hope to use our findings to construct guidelines for teachers creating their own content in the system, which can be embedded directly into the authoring tools teachers use, providing useful feedback on their problem design.

5.1 Randomized Controlled Trials

Having found a set of features that are associated with differences in student engagement and learning, our next step will be to conduct a set of randomized controlled trials (RCTs) to test whether the effects we found are genuinely causal, and whether re-designing problems based on these findings can improve student outcomes. By determining which of these features are causal, we can expand scientific understanding of learning and engagement in online learning systems. By developing methods for concretely improving math problems, we can develop better guidelines and recommendations for the many instructors (and others) developing problems for the ASSISTments platforms. In the longer-term, we hope to make all of the problems in the ASSISTments platform engaging and educationally effective for each of the growing number of students who use ASSISTments to learn mathematics and other subjects.

5.2 Continued Feature Engineering

Another important area of future work will be to conduct further feature engineering, particularly in terms of text features specific to the language of mathematics. One of the shortcomings of the current study is that the language of mathematics is poorly modeled in existing tools. In addition to challenges cause by domain or context-specific uses of certain words, many semantic taggers rely on syntactic probabilities that may be difficult to capture when math problems are interspersed with text. Simply developing taggers that can identify embedded mathematics formulas (e.g., labeling '3+2' as addition) could help to ameliorate this issue. We hope that, by developing more robust tools for the analysis of this particular corpus, we will be able to better predict and understand learning and engagement.

As research progresses, features derived from combinations of Wmatrix tags will also become important since many of the sub-categories within and across Wmatrix's lexical fields may be semantically similar enough, or co-occur frequently enough, to warrant combining them within ASSISTments data. For example, Wmatrix treats *deciding* as separate from *choosing, selecting*, and *picking,* but this division may not be useful in mathematics learning corpora. Likewise, feature combinations may help to

contextualize Wmatrix categories that are prone to incorrectly categorizing high-frequency words. For example, since many features in this study are highly correlated with M1, combinations involving this tag may be used to differentiate its use in instructions to students (e.g., "You have 3 attempts *left*") from its use in physical descriptions related to geometry (e.g., "Jill turns *left* and walks 3 more miles.").

5.3 Directions for Future Work

In this paper, we discovered relationships between semantic elements of text in the ASSISTments system and learning, affective, and behavioral student outcomes. In doing so, this work contributes to the emerging body of research studying the design of mathematics problems at scale.

Our findings show that a large number of semantically meaningful relationships exist, some of which correlate with a wide range of learner outcomes. These features provide insights that will help to develop guidelines for effective problem designs in ITSs. However, the existing suite of tools available for large scale textual analysis may not be optimal for tagging the specialized language of mathematics found in the ASSISTments system. Thus an additional area for future work includes the development of semantic taggers that are more appropriate for mathematics corpora. These efforts will help us to better understand how the linguistic properties of math problems influence student success at scale. In turn, by exploring potential relationships between persistence and student perceptions of challenge, we can work to design mathematics problems that are both more informative and more engaging.

6. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation (NSFDRL 1252297). Any opinions and findings expressed are the authors' and do not necessarily reflect the views of the NSF.

7. REFERENCES

[1] Adar, E., & Datta, S. Building a Scientific Concept Hierarchy Database (SCHBASE). *Ann Arbor*, *1001*, 48104.

[2] Baker, R. S., Goldstein, A. B., & Heffernan, N. T. (2011). Detecting learning moment-by-moment. International Journal of Artificial Intelligence in Education, 21(1-2), 5-25.

[3] Barber, C. (1962). Some measurable characteristics of modern scientific prose. *Contributions to English syntax and philology*, 21-43.

[4] Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. J. Royal Stat. Soc. B: Methodological, 289-300.

[5] Binder, J., Westbury, C, McKiernan, K, Possing, E, Medler, D. (2005). Distinct brain systems for processing concrete & abstract concepts. *J. Cog. Neurosci.*, *17*(6), 905-17.

[6] Carriedo, N., Elosúa, M., & García-Madruga, J. (2011). Working memory, text comprehension, and propositional reasoning: A new semantic anaphora WM test. *The Spanish J. Psych.*, *14*(01), 37-49.

[7] Cocea, M., Hershkovitz, A., & Baker, R.S. (2009). The impact of off-task and gaming behaviors on learning: immediate or aggregate?

[8] Cook, A. E., Myers, J. L., & O'Brien, E. J. (2005). Processing an anaphor when there is no antecedent. Discourse Processes, 39(1), 101-120.

[9] D'Mello, S., & Graesser, A. (2012). Dynamics of affective states during complex learning. *Learning & Instruction,* 22(2), 145-157.

[10] Drouin, P. (2010). Extracting a bilingual transdisciplinary scientific lexicon. eLexicography in the 21st C: new challenges, new applications. Louvain-la-Neuve: Presses Universitaires de Louvain/Cahiers du CENTAL, 43-53.

[11] Ferketich, S. (1991). Focus on psychometrics. Aspects of item analysis. Research in Nursing & Health, 14(2), 165-168.

[12] Forsyth, R., Ainsworth, S., Clarke, D., Brundell, P., & O'Malley, C. (2006). Linguistic-computing methods for analysing digital records of learning. In Online Proceedings of the 2nd International Conf. on e-Social Science, 28-30.

[13] Gacitua, R., Sawyer, P., & Rayson, P. (2008). A flexible framework to experiment with ontology learning techniques. Knowledge-Based Systems, 21(3), 192-199.

[14] Gowda, S.M., Pardos, Z.A., & Baker, R.S. (2012). Content learning analysis using the moment-by-moment learning detector. In Intelligent Tutoring Systems (pp. 434-443). Springer Berlin Heidelberg.

[15] Graesser, A. McNamara, D, Louwerse, M., & Cai, Z. (2004). Coh-Metrix: Analysis of text on cohesion and language. *Behavior research methods, instruments, & computers*, *36*(2), 193-202.

[16] Graesser, A., McNamara, D., Louwerse, M. (2012). Sources of text difficulty: Across the ages and genres. Sabatini & Albro, Assessing reading in the 21st C.: Aligning & applying advances in the reading & measurement sciences. Lanham, MD: R&L Education.

[17] Heffernan, N., & Heffernan, C. (2014). The ASSISTments Ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *Inter'l J. Artificial Intelligence in Ed.*, *24*(4), 470-497.

[18] Heiner, C., Beck, J., & Mostow, J. (2004). Improving the help selection policy in a Reading Tutor that listens. In *InSTIL/ICALL Symposium*.

[19] Jessen, F., Heun, R., Erb, M., Granath, D., Klose, U., Papassotiropoulos, A., & Grodd, W. (2000). The concreteness effect: Evidence for dual coding and context availability. *Brain & Language*, *74*(1), 103-112.

[20] Kittur, A., & Kraut, R. E. (2008). Harnessing the wisdom of crowds in wikipedia: quality through coordination. Proc. of ACM Conf. on Computer-supported cooperative work 37-46.

[21] Kittur, A., Chi, E. H., & Suh, B. (2008). Crowdsourcing user studies with Mechanical Turk. *Proc. of the ACM SIGCHI conference on human factors in computing systems* 453-456.

[22] Klin, C. M., Weingartner, K. M., Guzmán, A. E., & Levine, W. H. (2004). Readers' sensitivity to linguistic cues in narratives: How salience influences anaphor resolution. *Memory & Cognition*, *32*(3), 511-522.

[23] Kousta, S., Vigliocco, G, Vinson, D, Andrews, M., & Del Campo, E. (2011). The representation of abstract words: why emotion matters. *J. Exp. Psych: General*, *140*(1), 14.

[24] L'Homme, M. C. (2003). Capturing the lexical structure in special subject fields with verbs and verbal derivatives. A model for specialized lexicography. *International Journal of Lexicography*, *16*(4), 403-422.

[25] Light, L., & Capps, J. (1986). Comprehension of pronouns in young and older adults. *Developmental Psych.*, *22*(4), 580.

[26] Liu, Z., Pataranutaporn, V., Ocumpaugh, J., & Baker, R.S. (2013). Sequences of Frustration and Confusion, and Learning. In *EDM,* 114-120.

[27] Loginova, E., Gojun, A., Blancafort, H., Guégan, M., Gornostay, T., & Heid, U. (2012). Reference lists for the evaluation of term extraction tools. *Terminology & Knowledge Engineering (TKE'12)*, 30.

[28] Martin, J. H. (1994). METABANK: A KNOWLEDGE-BASE OF METAPHORIC LANGUAGE CONVENTIONS. *Computational Intelligence*, *10*(2), 134-149.

[29] McGregor, M. (1991). Language, culture and mathematics learning. McGregor & Moore (Eds.), *Teaching mathematics in the multicultural classroom: A resource for teachers and teacher educators*, 5-25. U. of Melbourne, School of Mathematics & Science Education.

[30] Morin, E., & Daille, B. (2010). Compositionality and lexical alignment of multi-word terms. *Language Resources and Evaluation*, *44*(1-2), 79-95.

[31] O'Toole, J. M. (1998). Climbing the fence around science ideas. *Australian Science Teachers Journal*, *44*(4), 51.

[32] Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., Heffernan, C. (2014) Population validity for Educational Data Mining models: A case study in affect detection. *British Journal of Educational Technology, 45* (3), 487-501.

[33] Ocumpaugh, J., Baker, R.S., Rodrigo, M.M.T., Salvi, A. van Velsen, M., Aghababyan, A., Martin, T. (2015). HART: The Human Affect Recording Tool. *Proc. of the ACM Special Interest Group on the Design of Communication (SIGDOC).*

[34] Pardos, Z. A., Baker, R. S., San Pedro, M., Gowda, S. M., & Gowda, S. M. (2014). Affective states and state tests: investigating how affect and engagement during the school year predict end-of-year learning outcomes. 1(1), 107-128.

[35] Paterson, D. G., & Tinker, M. A. (1946). Readability of newspaper headlines printed in capitals and in lower case. *Journal of Applied Psychology*, *30*(2), 161.

[36] Patterson, O. (2012). Automatic domain adaptation of word sense disambiguation based on sublanguage semantic schemata applied to clinical narrative (Dissertation, U. Utah).

[37] Pennebaker, J. W., Chung, C. K., Ireland, M., Gonzales, A., & Booth, R. J. (2007). *The development and psychometric properties of LIWC2007*. Austin, TX: LIWC. Net.

[38] Piao, S., Rayson, P., Archer, D., & McEnery, A. M. (2004). Evaluating lexical resources for a semantic tagger.

[39] Rayson, P. (2008). Wmatrix corpus analysis and comparison tool. Lancaster University.

[40] Schleppegrell, M. J. (2007). The linguistic challenges of mathematics teaching and learning: A research review. Reading & Writing Quarterly, 23(2), 139-159.

[41] Schutz, N. (2013). How specific is English for Academic Purposes? A look at verbs in business, linguistics and medical research articles. *Language and Computers*, *77*(1), 237-257.

[42] Toppino, T. C., & Ann Brochin, H. (1989). Learning from tests: The case of true-false examinations. *The Journal of Educational Research*, *83*(2), 119-124.

[43] van de Sande, B. (2013). Measuring the moment of learning with an information-theoretic approach. In EDM 288-291.

[44] Walkington, C., Clinton, V., & Howell, E. (2013). The associations between readability measures and problem solving in algebra. Martinez. & Castro Superfine, Eds. *Procs 35th meeting of the N. Am. Ch. Inter'al Group Psych. of Mathematics Ed. (pp.* 86-89). Chicago, IL: U. Ill, Chicago.

[45] Walkington, C., Clinton, V., Ritter, S. N., & Nathan, M. J. (2015). How readability and topic incidence relate to performance on mathematics story problems in computer-based curricula. *Journal of Educational Psychology*, *107*(4), 1051.

[46] Weld, D. S., Adar, E., Chilton, L., Hoffmann, R., Horvitz, E., Koch, M., ... & Mausam, M. (2012). Personalized online education—a crowdsourcing challenge. In Workshops at the 26th AAAI Conference on Artificial Intelligence (pp. 1-31).

[47] White, S. (2012). Mining the Text: 34 Text Features That Can Ease or Obstruct Text Comprehension and Use. Literacy Research and Instruction, 51(2), 143-164.

[48] Wiest, L. (2003). Comprehension of mathematical text. Philosophy of mathematics education journal, 17, 458.

[49] Baker, R.S.J.d., de Carvalho, A.M.J.A., Raspat, J., Aleven, V., Corbett, A.T., Koedinger, K.R. (2009) Educational Software Features that Encourage and Discourage "Gaming the System". *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, 475-482.

[50] Gibbons, J. D., & Chakraborti, S. (2011). *Nonparametric statistical inference* (pp. 977-979). Springer Berlin Heidelberg.

An Ensemble Method to Predict Student Performance in an Online Math Learning Environment

Martin Stapel
Department of Computer Science
Humboldt University of Berlin
Berlin, Germany
martin.stapel@hu-berlin.de

Zhilin Zheng
Department of Computer Science
Humboldt University of Berlin
Berlin, Germany
zhilin.zheng@hu-berlin.de

Niels Pinkwart
Department of Computer Science
Humboldt University of Berlin
Berlin, Germany
niels.pinkwart@hu-berlin.de

ABSTRACT

The number of e-learning platforms and blended learning environments is continuously increasing and has sparked a lot of research around improvements of educational processes. Here, the ability to accurately predict student performance plays a vital role. Previous studies commonly focused on the construction of predictors tailored to a formal course. In this paper we relax this constraint, leveraging domain knowledge and combining a knowledge graph representation with activity scopes based on sets of didactically feasible learning objectives. Specialized scope classifiers are then combined to an ensemble to robustly predict student performance on learning objectives independently of the student's individual learning setting. The final ensemble's accuracy trumps any single classifier tested.

Keywords

educational data mining, student performance prediction, ensemble methods, knowledge graph

1. INTRODUCTION

Performance prediction is one cornerstone of a fully personalized learning environment and also an important component of the efforts to deliver quality education. Higher education institutes, for example, are striving to incorporate predictive elements into their educational processes to better support students. Online systems like Massive Open Online Courses, Intelligent Tutoring Systems (ITSs) and increasingly Learning Management Systems (LMSs) also look for methods to compensate the lack of face-to-face interactions with teachers and the resulting problems with student's retention, completion, and graduation rates. Knowledge engineering and Educational Data Mining (EDM) methods and tools have helped to increasingly sharpen the models of student knowledge within these environments.

The foundations for performance prediction and student modeling were introduced more than four decades ago with Knowledge Tracing [1] and have since been constantly refined and extended to build diverse student models [3, 7, 17]. Such models are widely used in ITSs to allow for adaptive and personalized behavior. Technological advancements and innovations enabled the development of more elaborate online learning environments that reduce learning costs [8] and overcome space and time limitations. Through the use of such systems, previously inaccessible data about student's learning behaviors and their activities are now at hand. Analyzing student activities has become an important EDM task [2].

Data mining and machine learning approaches are often employed for the student performance prediction task since classification is one of the most frequently studied challenges by data mining and machine learning researchers. Such analyzes showed the ability to predict student's performance [15, 25] and even their drop out [14] in a broad range of educational technology environments. Usually, such prediction efforts are centered around a rather formal course students have to follow, like a university course or a structured online-only course. In this paper, we focus on a learning technology system that deliberately refrains from such a course structure.

This math learning system – called bettermarks – offers its users, students and teachers alike, guidance without imposing a course on them. The learning platform supports different curricula as well as flexible teacher interventions and leads students to a particular learning objective at their pace. The learning objectives range from introductory knowledge to advanced concepts. For our work in this blended K-12 learning environment where students either work in a traditional school setting or on their own, we opted to focus on performance data for the prediction task. We combine measured performance data with a knowledge graph representation of the platform's learning objectives, without the need for a strict course structure. Pursuing the prediction problem from this angle fully utilizes the math content organization and thereby directly connects extensive domain expertise and machine learning methods. The knowledge graph models how learning objectives are interconnected via pre-knowledge requirements. We use this graph to identify didactically feasible activity scopes. Based on those, special-

ized classifiers are trained and finally combined to predict student performance on a learning objective in an ensemble.

The remainder of this paper is organized as follows. In Section 2, we review how student modeling is approached in traditional ITSs and recent research on student performance prediction in different environments. Section 3 introduces the specific usage scenario of the bettermarks platform, its distinct characteristics, and the dataset. The following Section 4 describes our research method, including the generation of the classifier ensemble. Section 5 presents our findings and Section 6 concludes the paper with a discussion.

2. STATE OF THE ART

For Intelligent Tutoring Systems, student modeling is one major task which has been used for making assumptions about student's latent attributes. It uses observations of student's performance (e.g., correctness of given answers) or student's actions (e.g., the time a student spent on an exercise) to estimate student's hidden attributes, like knowledge, preferences or even motivational state. Which usually cannot be detected directly.

A well-established method for student modeling that has been used in various fashions for more than 40 years now is called Knowledge Tracing (KT). This technique was pioneered by Atkinson [1] and substantially developed by Corbett and Anderson. Their variant is based on a 2-state dynamic Bayesian network [7]. The observed variable is the student performance, and the student knowledge is the latent one which is estimated. Regarding student performance, there are two additional parameters to account for accidental and careless mistakes (slip) and solving an exercise despite not knowing (guess). The set of parameters is completed with one for any prior knowledge a student might already have and one for her learning rate. This standard KT model is often used for its abilities to provide skill level diagnostics. In recent years, a range of extensions to Knowledge Tracing have been proposed to mitigate some of its shortcomings. A particularly noteworthy one is Baker et al.'s contextual guess and slip model [3]. Recently, Pardos and Heffernan proposed an extension to the standard model to incorporate item-level difficulty [17].

Besides KT, other approaches exist. A comparably new option is called Performance Factor Analysis (PFA) which was proposed by Pavlik et al. [19]. Their student modeling method uses a logistic regression model with a reconfigured version of Learning Factor Analysis [6] whose skill variable is replaced by one parameter per item (e.g., exercise, question, knowledge component) and the student variable is dropped entirely. The model estimates the individual item difficulty as well as effects of prior successes and failures for each skill. It predicts student performance based on item difficulty and prior performances. Comparative analyzes of KT's and PFA's performance showed that either of them appear to be suitable for student modeling [4, 10, 19].

In learning environments without such semantically rich data and a domain model, data mining, and machine learning approaches are often applied for the performance prediction task. The goals here remain mainly the same, with additional emphasis on early warning and drop out prediction. In general, student's prior performances are used to train different machine learning models to predict future test or exam performance, similarly to PFA. However, not all environments provide access to performance data. The steadily growing number of LMSs, for instance, do not always collect such data. In such environments, one has to resort to data about student's activities. Hu et al. developed an early warning system based on student's usage of an LMS utilizing metadata captured while students interact with the system [12]. The studied dataset includes information like login counts, time spent logged in, and metadata concerning homework assignments and was gathered during two semesters of a fully online university course with 300 enrolled students. The course required students to attend online classes and watch videos in specific time periods. To build their early warning system, the authors generated three datasets to create different periods to study (4, 8 and 13 weeks) and applied three often used classification techniques, C4.5, CART, and logistic regression. Additionally, Hu et al. employed AdaBoost to achieve greater prediction accuracy which led to the best performing classifier constructed from AdaBoost and CART. This classifier achieved a prediction accuracy of at least 0.972 on each of the three datasets. A similar scenario, yet more open, was studied by Zacharis who investigated student performance related to online activities in an LMS, which was used as part of a blended learning university course [29]. 134 students were enrolled in this course for one semester. To account for student-teacher and student-student interactions which could not be observed, all of the captured online activities were treated equally while searching for significant correlations with the student's final grades. Out of 29 variables, almost 50% were found to be important. A stepwise regression yielded a model with four variables which were used in a logistic analysis to discriminate between failing and not at risk students. An overall classification accuracy of 81.3% was achieved. Predicting student performance in a timely fashion as done by Koprinska et al. underscores the usefulness of performance data [13]. Their studied dataset included submission sets, assessment information, and engagement data from a discussion forum. All of the data was gathered from different online systems used in a blended university course. Koprinska et al. defined their classification problem as a three class problem and divided the 224 participating students into high-, average- and low-level students based on exam performance at the end of the course. To predict the exam result, they employed a decision tree classifier which achieved an accuracy score of 72.69% using the complete course data. Using just the data from the first half of the course led to an accuracy score of 66.52%. Here, almost half of the used features are performance related.

Our work uses a similar approach to predict student performance in a blended K-12 learning environment. The critical difference between other datasets used in previous research and ours is that students on the bettermarks platform neither attend nor follow a formal course. The system provides teachers and students with "math books" for a term's curriculum. Since the learning platform is often used supplementary to traditional lessons in class, teachers make use of the learning material at their discretion. Likewise, students in a self-regulated learning setting might pick a couple of learning objectives or decide to work through a whole

book on their own. The resulting freedom for students and teachers introduces a huge amount of diversity in the user behavior and poses challenges for performance prediction algorithms. To fully capture student behavior and overcome the problem of fitting a single prediction model based on diverse data sources, Essa and Ayad proposed a domain-specific decomposition of different (online) learning related aspects [9]. The final prediction would consequently consist of an ensemble of classifiers specialized on each aspect's data. Hence, the resulting model should be more generalizable and flexible than models build on single courses. Building on this idea, we focused on learning objectives as the common data underlying every user's interaction and decomposed the math content organization of the platform into different activity scopes. Classifiers trained on those scopes act as base classifiers for the developed ensemble which robustly predicts student performance independently of their learning situation.

The particularly chosen focus on exercises (or learning objectives, for that matter) in our research is a crucial distinction to prior ensemble-based prediction works. Student performance within an ITS as well as on a paper post-test was predicted by Baker et al. utilizing ensembles of different student models (including the previously discussed BKT and PFA). The achieved results let the authors conclude that ensembling appeared to be only slightly better [4]. Looking further into the previous results and concentrating exclusively on post-test predictions did not yield better prediction results over the best individual models [18]. Again, different student modeling approaches were combined to ensembles. Gowda et al. found that ensembles build on large enough datasets (about 15 times more data than used in the previous two studies) can very well yield superior prediction performance, even with similar models as a base [11].

3. THE USAGE SCENARIO

The bettermarks system is an online math learning platform with more than 100k interactive exercises, covering K-12 math curricula (grades 4-10) in English, Spanish, German and Dutch language. It is designed to be used in math classes at school without implying a formal course structure. Teachers can decide to teach math entirely with the system, supplement their lessons with related bettermarks content right in class, or assign exercises as homework. At any time, teachers can be aware of their student's progress through detailed reports which present high-level performance aggregation as well as every single solution attempt. The system can also support and guide students working on their own in a self-regulated learning setting without additional teacher interventions. Each month, more than 100k students across Europe and America use bettermarks.

Besides offering detailed textbook-like explanations of math topics, the primary means of learning math on the bettermarks platform are math exercises. Exercises are grouped into exercise series. Each series helps students achieve a well defined and fine-grained learning objective. Examples of such learning objectives are "Calculate the surface area of a prism given the edge lengths and the height" or "Find the zeros of linear and quadratic functions." These series are arranged into digital books based on curricular themes and didactical concepts without imposing any curriculum

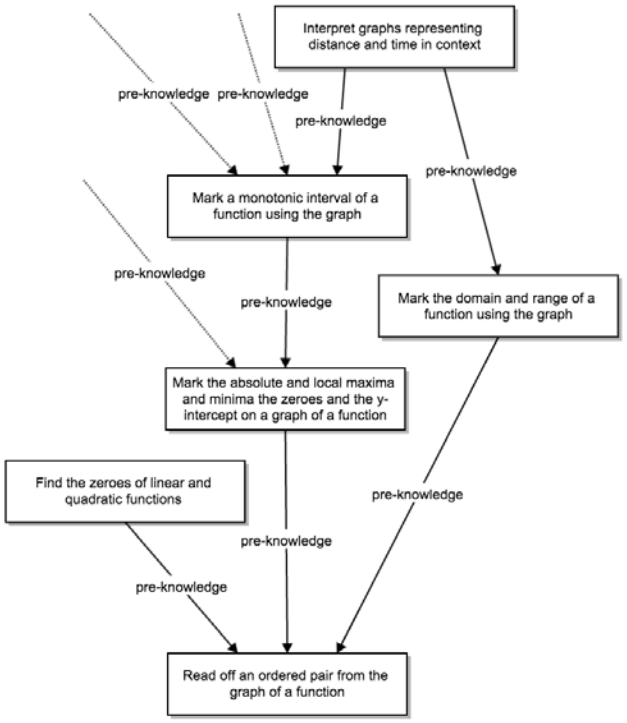


Figure 1: Small section of the entire knowledge graph spanning more than 1,500 vertices

structure on the user. Each book is organized similarly to a printed math book with chapters and series of exercises within these chapters. Behind those books that are visible to teachers and students lies a knowledge graph (not visible to users). This graph describes how learning objectives relate to each other regarding required prior knowledge.

3.1 Knowledge Graph

The idea of a concept map was first introduced in the 1970s by Novak. In his later work, he used this framework to organize and connect already acquired knowledge with new knowledge [16]. The usefulness of maps related to the original ideas for learning and assessment in technology-based learning environments has already been shown [24, 27]. Building on these concepts, the underlying structure of the bettermarks content is called a knowledge graph. This graph is built by connecting nodes concerning their pre-knowledge requirements. Each of the graph nodes represents a learning objective – a particular skill a student reaches once she successfully finishes a series of exercises designed especially for this skill. These objectives include introductory/elementary skills as well as core knowledge and advanced skills. The direction of an edge indicates which node is defined as the required pre-knowledge for another node. A particular node might have more than one pre-knowledge node. The entire bettermarks knowledge graph contains more than 1,500 learning objectives in total. A small subset of them is shown in Figure 1. A digital math book on the bettermarks platform includes a number of these learning objectives. Usually, not all of them are directly (or indirectly) related.

3.2 Data

The analysis in this paper focuses on the particularly well-frequented book "Calculating Percents" from the German version of the bettermarks system. From this book's learning objectives, we chose one with a relatively large amount of required pre-knowledge as a classification target. It is called "Calculate decreased and increased base values in context" and located close to the end of the book. The data was gathered during the entire year of 2015 and includes student's activities on the bettermarks platform 40 days before their first attempt on the classification target. The 40 day period allows students in a school setting to reasonably work their way to this objective. In total, the dataset includes performance measurements of 566 students on 903 different learning objectives which are the results of 10,363 solution attempts by 6th - 10th-grade students from all over Germany. A student is free to repeat an exercise series as often as she wants. Since the system presents the student's best solution attempt to a teacher first, we also used this result for each student and learning objective. Table 1 shows a randomly chosen sample of the entire dataset with results on three learning objectives (represented by identifiers). The results correspond to the ratio of correctly solved exercises in a series. It is evident that not all learning objectives have been addressed by the same amount of attempts. The last column shows the highest success rate on the classification target achieved by a student within 3 hours of starting the exercise series for the first time. We noticed that students employed different strategies involving repetitions while solving exercise series which makes the success rate achieved in the first attempt a bad indicator for the final result a student settles on by continuing with the next series. Therefore, the 3 hours allow students some time to repeat the exercise series and also account for the fact that students might have reached the classification target during their math lesson at school and want to repeat the exercise series again at home. These collected performance measurements are used as possible features in our classification models.

4. RESEARCH METHODOLOGY

Over the course of the following section, our research method is discussed in detail, we were guided by a two-fold research focus: (1) Can an ensemble of classifiers based on the decomposed math content organization accurately predict student performance? (2) Given the usage scenario, is this approach suitable for an "early prediction" setting? Since the bettermarks system offers its users lots of flexibility, an early prediction task is different from a formal course's early prediction task. In our case, the early prediction challenge is not transferable to a subset of the course's allocated time and exercises. Instead, we looked into students showing low usage rates over the examined period. In our case (and in contrast to online-only environments), a lack of activity does not imply that students did not attend a regular math lesson and progressed in school.

In a first step, the math content was decomposed into activity scopes relating to the classification target. A following pre-processing step used different aggregations to gain better insights into the available dataset. The primary concerns that governed this step refer to how much of the data is missing and if the classifiers can learn from roughly balanced classes created by the class split. The first question is also relevant regarding the number of actually achieved learning objectives by students within the different scopes since those directly translate into the initial feature sets. Afterwards, six different algorithms were evaluated on each scope as base classifiers for the ensemble. The process is described in the Ensemble Construction section which also discusses the imputation and standardization strategies we employed. Following the final model selection, the ensemble's weights were optimized. This step also concluded the generation of the entire ensemble.

4.1 Activity Scopes

To reflect the flexibility the learning system offers its users, we defined three activity scopes and constructed specialized classifiers for them. All scopes center around a particular subset of the knowledge graph's vertices and thus decompose the graph into relevant groups related to the classification target. The subgraph spun by the classification target's vertice via the pre-knowledge relation serves as the binding element between the three scopes.

The first scope includes all learning objectives that are part of the classification target's pre-knowledge in the knowledge graph. These are all vertices connected directly or indirectly to the classification target through pre-knowledge relation edges. In total, those are 35 different learning objectives for our chosen classification target "Calculate decreased and increased base values in context."

The classification target is located in the math book "Calculating Percents". This book with all of its learning objectives creates the second activity scope, the math book scope. Excluding the classification target itself, the set of potential features for this scope contains 24 learning objectives. Since the book was created with didactical considerations in mind, the math book's learning objectives are arranged similarly to the knowledge graphs vertices. Still, this scope and the pre-knowledge scope share only five learning objectives.

The final scope includes student's activities on learning objectives that are not part of the math book's scope. All of these learning objectives are part of the knowledge graph as well, but those are located in other math books. Nevertheless, the resulting set was not partitioned any further by their books. This scope could share up to 30 learning objectives with the first scope but does not include any from the book's scope. Those would be the learning objectives the pre-knowledge scope does not share with the book's scope. The actual number depends entirely on the student's activities during the examined period. With these defined scopes we attempted to model the different paths teachers and students might have taken to approach the classification target.

4.2 Pre-processing

In Germany, the bettermarks system is often used in math classes to supplement regular lessons. Therefore, it is not expected that students solve a vast amount of exercise series over the chosen 40 days. Figure 2 shows that the median of different exercise series per student is at 14.5 series with the 0.75 percentile at 23 series.

This result suggests that the amount of gathered performance measures per learning objective could be rather sparse

Table 1: Sample of user IDs with success rates on different learning objectives

user_id	Learning objectives				classification_target
	PruZiPruZiRFo.LOB04	PruZiPruZiRDr.LOB06	ZUZUProp.LOB01	...	
369947		0.333		...	0.675
92083	0.708	0.333		...	0.921
5625246	0.708	0.333	0.429	...	0.447
347284	0.208	0.500		...	0.475
361389	0.417	0.333		...	0.675

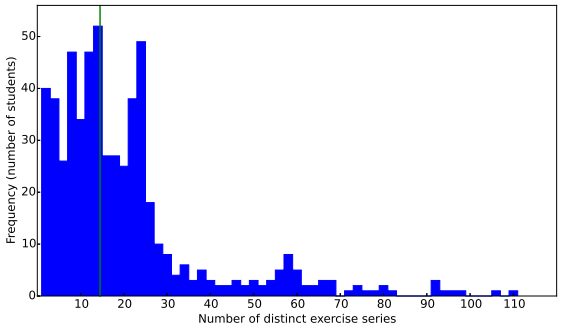


Figure 2: Students solve a rather small number of different series with the median at 14.5 series (indicated as green vertical line)

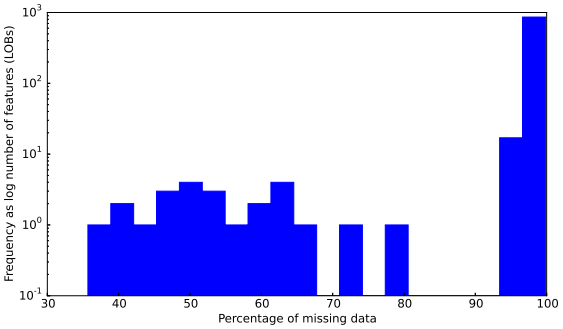


Figure 3: Data Sparsity

for the majority of series. In fact, 566 students worked on 903 different learning objectives with an average of almost 20 different series per student. Further examination reveals that only 22 learning objectives had up to 70% of the data missing. The data sparsity is illustrated in Figure 3. It is important to employ a suitable data imputation strategy and apply feature selection means during the construction of the different classifiers later to cope with this sparse dataset.

We decided to split the classes at a success rate of 0.75. One class is composed of students with success rates lower than 0.75, whereas the second one contains students with success rates of at least 0.75 which would translate to a separation of top performing students from all other students. This class split has the benefit of dealing with quite balanced classes. Figure 4 shows the median success rate at 0.76 (red) and our class split slightly left to it at 0.75 (green). The resulting spread is 45.6% to 54.4% between both classes.

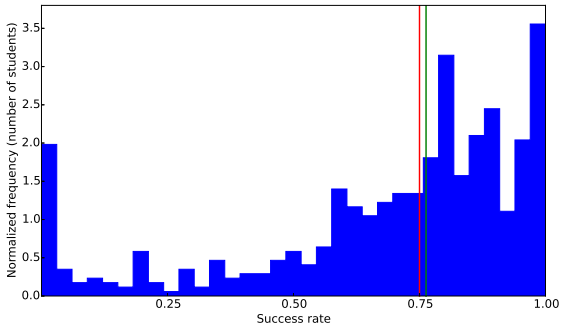


Figure 4: Measured success rates at the classification target. The red line indicates class split at 0.75 and the green one the median success rate at 0.76

The dataset does not contain the entire set of pre-knowledge learning objectives. Out of 35 possible learning objectives, only data for 16 is present. One possible explanation is that pre-knowledge learning objectives are not always part of a single term's curriculum (but available for teachers to choose from). Hence, it is not expected that students work their way through the entire pre-knowledge of a particular learning objective during a short period. All of the expected 24 book scope's objectives are present in the dataset.

4.3 Ensemble Construction

An ensemble of classifiers blends predictions from multiple models with a two-fold goal: The first intent is to boost the overall prediction accuracy compared to a single classifier. The second benefit is a better generalizability due to different specialized classifiers. As a result, an ensemble can find solutions where a single prediction model would have difficulties. A key rationale is that an ensemble can select a set of hypotheses out of a much larger hypothesis space and combine their predictions into one [22].

For our purposes, we started with a set of well-known classification algorithms and used nested cross-validation to determine their performance. The algorithm with the highest average accuracy score in each scope is afterwards chosen for final model selection. The performance of the best model was evaluated on a hold-out dataset (30% of the entire data). Once the model selection took place, the weights for the ensemble were adjusted, again, with cross-validation and the final ensemble's performance evaluated on the hold-out dataset. The following sections describe the whole process in detail.

Table 2: Average accuracy achieved in nested cross-validation for each tested algorithm and scope

Algorithm	Book	Pre-knowledge	Outside
Decision Tree with AdaBoost	**0.715**	0.634	0.525
k-Nearest Neighbors	0.629	0.609	0.546
Logistic Regression	0.682	**0.659**	0.538
Naïve Bayes	0.654	0.636	0.467
Random Forest	0.679	0.652	**0.550**
Stochastic Gradient Descent	0.624	0.594	0.525

4.3.1 Selecting Algorithms

A set of six commonly used classification algorithms were chosen as potential base models. The set consists of Random Forest, Decision Tree with AdaBoost, Logistic Regression, k-Nearest Neighbors, Stochastic Gradient Descent and a Naïve Bayes implementation. For each scope, a classification pipeline was created.[1] To impute missing data we opted for filling missing values with the mean success rate of the particular feature. Tests with the median and the mode did not significantly influence later on achieved classification results. The data was robustly standardized by removing the median and scaling the data according to the Interquartile Range (IQR)[2]. Each pipeline used a scope-specific variance threshold on the imputed data as feature selection mechanism. The actual threshold is determined during model selection (0-60% of the feature's variance). The purpose is to remove features that do not meet the set threshold. This applies to features with low variance due to rather uniform student activities as well as to features with large amounts of imputed data.

To get a conservative and thus fairly unbiased base estimate of each classifiers performance [26], we used nested stratified cross-validation with 10 folds on the outside and 5 folds on the inside with randomized search [5] over the parameter space. Depending on the algorithm, the search space was limited to reasonable values such as restricting the number of trees in a forest. The search included 100 sets of candidate parameters. Table 2 shows the results for each classification algorithm and scope. The best performing algorithm is highlighted in each column.

4.3.2 Model selection and Ensemble construction

AdaBoost on Decision Tree for the math book scope, Logistic Regression for the pre-knowledge scope and Random Forest for the outside scope were picked for the final model selection. It was done by 10-fold cross-validation and a random search over 750 sets of candidate parameters. The best performing model of each scope was afterwards chosen and re-trained on the entire training set for the ensemble.

[1]The pipeline facility, as well as the used algorithms' implementations are part of scikit-learn [20].
[2]The IQR is the range between the 1st quartile (0.25 percentile) and the 3rd quartile (0.75 percentile)

Table 3: Prediction accuracy on the test set

Classifier	Prediction accuracy
Baseline	0.594
Pre-knowledge scope	0.682
Book scope	0.705
Outside	0.647
Ensemble	**0.735**

To construct the ensemble we opted for a soft voting strategy rather than using hard voting. A soft voting strategy has the significant advantage of weighing the three scopes differently. The alternative would be to use a majority decision among the three classifiers where each classifier's vote weights equally. Instead, the ensemble uses soft voting to classify students based on the argmax of the sums of each classifier's predicted probabilities. To determine the weights to be associated with each classifier, we used random search with 10-fold cross-validation on 3k parameter sets. The emerged ensemble with tuned weights was then tested on the hold-out part of the dataset.

5. RESULTS

To assess the performance of each classifier as well as of the entire ensemble more thoroughly we also added a baseline classifier. This simple classifier always predicts the majority class. Table 3 shows each classifier's prediction accuracy on the hold-out dataset.

As before with the nested cross-validation results, the accuracy ranking over the three scopes stayed the same – the book scope's classifier performed best (0.705) followed by the pre-knowledge scope's classifier (0.682). With a prediction accuracy of 0.594, the baseline classifier scores below all other approaches. The constructed ensemble achieved the best prediction accuracy with 0.735.

Since the ensemble showed an improved accuracy on the test set, we investigated the remaining classification errors further. Table 4 displays the confusion matrix for the book scope's classifier which is the best single-scope classifier. As a comparison, Table 5 shows the confusion matrix for the final ensemble. Out of the two, the latter made slightly more errors of type I. This is especially unfortunate because in our case, false positive errors translate to students incorrectly classified as top performers even though they could not reach the required success rate threshold. In our setting, errors of this type are arguably more expensive than classification errors of type II where a student would be wrongly classified as a low scoring student. If our prediction method would be used to trigger human interventions a teacher might determine rather quickly if a student is able to pass a test or not. However, if the system fails to notify the teacher in the first place, she might not at all be aware of a potential problem with the student's performance. Thus, the problem would be revealed after the student has aleardy failed.

Table 4: Book scope classifiers's confusion matrix

	Other students	Top performers
Other students	51	18
Top performers	32	69

Table 5: Ensemble's confusion matrix

	Other students	Top performers
Other students	50	19
Top performers	26	75

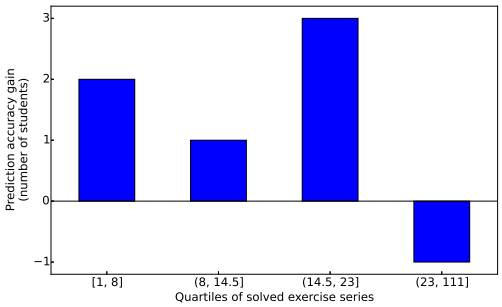


Figure 5: Ensemble's accuracy gain over book scope's classifier per quartile

Lastly, to assess the ensemble's ability to accurately predict student performance in an early prediction task, the accuracy of the best single-scope classifier and the ensemble was compared based on quartiles of student's number of solved exercise series. As described above, 50% of the students in our dataset solved up to 14.5 different exercise series in the examined period. To be used effectively in an early prediction setting, a suitable classifier needs to be able to accurately predict the right class with few data points. Figure 5 shows the accuracy difference between the book scope's classifier and the entire ensemble for each quartile. In the first three quartiles the ensemble predicts more students correctly than the book scope's classifier. These results lead to the conclusion that our approach has the potential be used in an early prediction setting.

6. DISCUSSION AND OUTLOOK

We investigated an approach that decomposes the math content structure underlying an online math learning platform, trains specialized classifiers on the resulting activity scopes and uses those classifiers in an ensemble to predict student performance on learning objectives. Students using this particular math learning platform achieve learning objectives without a formal course imposed on them which is quite different from course-centered online-only or blended learning environments. We showed that looking closer at the math exercises helped us build a robust classification model that can cope with student's notably diverse behavior due to the lack of a strict course framework. Using the knowledge graph to decompose the content domain enabled the individual prediction models to better grasp nuances of student's activities.

In general, the results suggest that our approach yields a robust performance prediction setup that can correctly classify 73.5% of the students in the dataset. This is an improvement over every other classification approach we tested in our study. Further examinations revealed that the ensemble also outperforms the best single-scope classifier in an early prediction or early warning setting. Students with lower levels of activity would benefit the most from our ensemble approach since it clearly improves the prediction accuracy for those students, as we have shown. However, the increased prediction accuracy came with a price: a slight increase in false positives where students are wrongly classified as top performing students. Especially in our area of research, false positive errors like this should be reduced as much as possible if we want to improve educational processes and make a lasting impact on every stakeholder.

Looking closer at the classification errors, we found that in 12 cases the three scope classifiers unanimously attributed the wrong class to a student. Hence, the ensemble was not able to predict the class for these students correctly either. The reason is a shortcoming of the ensemble's soft voting strategy which cannot overturn matching predictions among its base classifiers. Rather than using a simple weighted ensemble, it is possible to use stacking and thus introduce a second stage classifier. This classifier takes the prediction results of the ensemble's base classifiers and employs them as features to predict the final class. The whole concept is known as stacked generalization and exists in different flavors [28]. Gowda et al. have already shown the significant benefits of more sophisticated ensemble methods in a prediction task [11]. Additionally, a number of different ensemble generation methods can be utilized to achieve better diversity within the base classifiers [21]. Besides extending the final ensemble with stacking and exploring the resulting benefits, our future work will include more performance related data, like the number of attempts or the total time a student has spent on a particular exercise series. These efforts will go hand in hand with additional feature selection strategies, and dimensionality reduction means to capture more scope-related nuances of student's performances.

We also plan to investigate whether student's diverse sequences of learning objectives can be used to improve feature extraction and selection. Scheiter and Gerjets' results regarding the order of presented problems and performance improvements point to a possible connection [23].

While some of the discussed extensions seem obvious, the most important challenge is to develop our approach into a strategy suitable for any learning objective in this scenario. Our current approach uses a narrow set of learning objectives and a specifically tailored ensemble. These constraints reduce the cold start problem but require a good strategy to cope with missing data, as we have described. Nevertheless, the ensemble cannot easily be repurposed at scale. Hence, investigating different strategies leading to a broadly applicable solution will be our primary focus.

References

[1] R. C. Atkinson. Ingredients for a theory of instruction. *American Psychologist*, 27(10):921, 1972.

[2] R. S. J. d. Baker. *Data Mining*, pages 112–118. Elsevier, 2010.

[3] R. S. J. d. Baker, A. T. Corbett, and V. Aleven. *More Accurate Student Modeling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowl-*

edge Tracing, pages 406–415. Springer Berlin Heidelberg, 2008.

[4] R. S. J. d. Baker, Z. A. Pardos, S. M. Gowda, B. B. Nooraei, and N. T. Heffernan. *Ensembling Predictions of Student Knowledge within Intelligent Tutoring Systems*, pages 13–24. Springer Berlin Heidelberg, 2011.

[5] J. Bergstra and Y. Bengio. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 13(1):281–305, 2012.

[6] H. Cen, K. Koedinger, and B. Junker. *Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement*, pages 164–175. Springer Berlin Heidelberg, 2006.

[7] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278.

[8] D. J. Deming, C. Goldin, L. F. Katz, and N. Yuchtman. Can online learning bend the higher education cost curve? *American Economic Review*, 105(5):496–501, 2015.

[9] A. Essa and H. Ayad. Student Success System: Risk Analytics and Data Visualization Using Ensembles of Predictive Models. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 158–161. ACM Press, 2012.

[10] Y. Gong, J. E. Beck, and N. T. Heffernan. *Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures*, pages 35–44. Springer Berlin Heidelberg, 2010.

[11] S. M. Gowda, R. S. J. d. Baker, P. Zachary A, and N. T. Heffernan. The sum is greater than the parts: ensembling models of student knowledge in educational software. *SIGKDD Explor. Newsl.*, 13(2):37–44, 2012.

[12] Y.-H. Hu, C.-L. Lo, and S.-P. Shih. Developing early warning systems to predict students' online learning performance. *Computers in Human Behavior*, 36:469–478, 2014.

[13] I. Koprinska, J. Stretton, and K. Yacef. Predicting Student Performance from Multiple Data Sources. In *Artificial Intelligence in Education: 17th International Conference, AIED 2015, Madrid, Spain, June 22-26, 2015. Proceedings*, pages 678–681. Springer International Publishing, 2015.

[14] I. Lykourentzou, I. Giannoukos, V. Nikolopoulos, G. Mpardis, and V. Loumos. Dropout prediction in e-learning courses through the combination of machine learning techniques. *Computers & Education*, 53(3): 950–965, 2009.

[15] L. P. Macfadyen and S. Dawson. Mining lms data to develop an "early warning system" for educators: A proof of concept. *Computers & Education*, 54(2):588–599, 2010.

[16] J. Novak. Clarify with concept maps. *The Science Teacher*, 58(7):44, 1991.

[17] Z. A. Pardos and N. T. Heffernan. *KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model*, pages 243–254. Springer Berlin Heidelberg, 2011.

[18] Z. A. Pardos, S. M. Gowda, R. S. J. d. Baker, and N. T. Heffernan. Ensembling predictions of student post-test scores for an intelligent tutoring system. In *EDM*, pages 189–198, 2011.

[19] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis – a new alternative to knowledge tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*, pages 531–538. IOS Press, 2009.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] A. Rahman and S. Tasnim. Ensemble classifiers and their applications: A review. *International Journal of Computer Trends and Technology*, 10(1):31–35, 2014.

[22] L. Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2009.

[23] K. Scheiter and P. Gerjets. The impact of problem order: Sequencing problems as a strategy for improving one's performance. In *24th Annual Conference of the Cognitive Science Society*, pages 798–803. Erlbaum, 2002.

[24] D. L. Trumpower, M. Filiz, and G. S. Sarwar. Assessment for Learning Using Digital Knowledge Maps. In *Digital Knowledge Maps in Education: Technology-Enhanced Support for Teachers and Learners*, pages 221–237. Springer New York, 2014.

[25] S. Valsamidis, S. Kontogiannis, I. Kazanidis, T. Theodosiou, and A. Karakos. A Clustering Methodology of Web Log Data for Learning Management Systems. *Educational Technology & Society*, 15(2):154–167, 2012.

[26] S. Varma and R. Simon. Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7:91, 2006.

[27] K. Weinerth, V. Koenig, M. Brunner, and R. Martin. Concept maps: A useful and usable tool for computer-based knowledge assessment? a literature review with a focus on usability. *Computers & Education*, 78:201–209, 2014.

[28] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

[29] N. Z. Zacharis. A multivariate approach to predicting student outcomes in web-enabled blended learning courses. *The Internet and Higher Education*, 27:44–53, 2015.

Predicting Post-Test Performance from Online Student Behavior: A High School MOOC Case Study

Sabina Tomkins[1], Arti Ramesh[2], Lise Getoor[1]
[1]University of California, Santa Cruz
[2]University of Maryland, College Park
satomkin@ucsc.edu, artir@cs.umd.edu, getoor@soe.ucsc.edu

ABSTRACT

With the success and proliferation of Massive Open Online Courses (MOOCs) for college curricula, there is demand for adapting this modern mode of education for high school courses. Online and open courses have the potential to fill a much needed gap in high school curricula, especially in fields such as computer science, where there is shortage of trained teachers nationwide. In this paper, we analyze student post-test performance to determine the success of a high school computer science MOOC. We empirically characterize student success by using students' performance on the Advanced Placement (AP) exam, which we treat as a post test. This post-test performance is more indicative of long-term learning than course performance, and allows us to model the extent to which students have internalized course material. Additionally, we analyze and compare the performance of a subset of students who received in-person coaching at their high school, to those students who took the course independently. This comparison provides better understanding of the role of a teacher in a student's learning. We build a predictive machine learning model, and use it to identify the key factors contributing to the success of online high school courses. Our analysis demonstrates that high schoolers can thrive in MOOCs.

Keywords

online education, high school MOOCs, student learning

1. INTRODUCTION

Massive Open Online Courses (MOOCs) have emerged as a powerful mode of instruction, enabling access around the world to high quality education. Particularly for college curricula, MOOCs have become a popular education platform, offering a variety of courses across many disciplines. Now open online education is being deployed to high schools worldwide, exposing students to vast amounts of content, and new methods of learning. Even as the popularity of high school MOOCs increases, their efficacy is debated [8]. One challenge is that the large amount of self direction MOOCs require may be lacking in the average high school student.

To understand the applicability of the MOOC model to high schoolers, we analyze student behavior in a year-long high school MOOC on Advanced Placement (AP) Computer Science. This course is distinguished from traditional college-level MOOCs in several ways. First it is a year-long course, while college MOOCs average 8-10 weeks in duration. This provides ample opportunity to mine student interactions for an extended period of time. Secondly, while traditional MOOCs have no student-instructor interaction, the high school MOOC that we consider incorporates instructor intervention in the form of coaching and online forum instructor responses. Evaluating the effectiveness of this hybrid model allows us to investigate the effect of human instruction on high school students, a group which may particularly benefit from supervision.

Finally, we introduce a post test as a comprehensive assessment occurring after the termination of the course. A valid post test should assess students' knowledge on critical course concepts, such that students' course mastery is reflected in their post-test score. We treat the Advanced Placement (AP) exam as a post test and consider students' performance on this test as being indicative of long term learning. Previous MOOC research evaluates students on course performance [4]. While course performance can be a good metric for evaluating student learning in the short term, post-test performance is a more informative metric for evaluating long-term mastery.

We propose and address the following research questions, aimed at evaluating the success of MOOCs at the high school level.

1. Can high school students learn from a MOOC, as evidenced here by their post-test (AP exam) performance?
2. How does coaching help students achieve better course performance and learning?
3. How can we predict student's post test performance from course performance, forum data, and learning environment?

Our contributions in this paper are as follows:

1. We perform an in-depth analysis of student participation and performance to evaluate the success of MOOCs at the high school level. To do so, we identify two course success measures: 1) course performance scores, and 2) post-test performance scores.

2. We evaluate the effect of two important elements of this high school MOOC: discussion forums and coaching, on student performance.
3. We use a machine learning model to predict student post test scores. First constructing features drawn from our analysis of student activities, then determining the relative predictive power of these features. We show that this process can be used to draw useful insights about student learning.

2. RELATED WORK

Research on online student engagement and learning, is extensive and still growing Kizilcec et al. [5], Anderson et al. [1], and Ramesh et al. [11] develop models for understanding student engagement in online courses. Tucker et al. [13] mine text data in forums and examine their effects on student performance and learning outcomes. Vigentini and Clayphan [14] analyze the effects of course design and teaching effect on students' pace through online courses. They conclude that both the course design and the mode of teaching influence the way in which students progress through and complete the course. Simon et al. [12] analyze the impact of peer instruction in student learning.

Particularly relevant to our findings is the impact of gaming the system on long-term learning. Baker et al. [2] investigate the effect of students gaming an intelligent tutor system on post-test performance. In the high school MOOC setting, we observe a similar behavior in some students achieving high course performance, but low post-test performance. We identify plausible ways in which these students can be gaming the system to achieve high course performance and present analysis that is potentially useful for MOOC designers to prevent this behavior.

There is limited work on analyzing student behavior in high school MOOCs. Kurhila and Vihavainen [6] analyze Finnish high school students' behavior in a computer science MOOC to understand whether MOOCs can be used to supplement traditional classroom education. Najafi et al. [9] perform a study on 29 participating students by splitting them into two groups: one group participating only in the MOOC, and another group is a blended-MOOC that has some instructor interactions in addition to the MOOC. The report that students in the blended group showed more persistence in the course, but there was no statistically significant difference between the groups' performance in a post-test. In our work, we focus on empirically analyzing different elements of a high school MOOC that contribute to student learning in an online setting. We use post-test scores to capture student learning in the course and examine the interaction of different modes of course participation with post-test performance. Our analysis reveals course design insights which are helpful to MOOC educators.

3. DATA

This data is from a two-semester high school Computer Science MOOC, offered by a for-profit education company. The course prepares students for College Board's Advanced Placement Computer Science A exam and is equivalent to a semester long college introductory course on computer science. In this work, we consider data from the 2014-2015 school year for which 5692 students were enrolled.

The course is structured by terms, units, and lessons. Lessons provide instruction on a single topic, and consist of video lectures and activities. The lessons progress in difficulty beginning with printing output in Java, and ending with designing algorithms. Each lesson is accompanied with activities. These activities are not graded, instead students receive credit for attempting them. Students take *assessments* in three forms: assignments, quizzes, and exams, each released every two weeks.

At the end of the year students take an Advanced Placement (AP) exam. Students can use their AP exam performance exam as a substitution for a single introductory college course. The AP exam score ranges from 1 to 5. In all, we have data for 1613 students who take the AP exam. This number is a lower limit on the total number of students who may have taken the course and the AP. The course provides a forum service for students, which is staffed with paid course instructors. Approximately, 30% of all students who created course accounts also created forum accounts, 1728 students in all.

This course is unique in that it provides a coach service which high schools can purchase. This option requires that the school appoint a coach, who is responsible for overseeing the students at their school. The coach is provided with additional offline resources, and has access to a forum exclusive to coaches and course instructors. The average classroom size is approximately 9 students with a standard deviation of approximately 12 students. The largest classroom size coached by a single coach is 72, while some coaches supervise a single student. Of all students who have enrolled in the course, approximately, 23% (1290) are coached and 77% (4402) are independent. From here on we refer to the students enrolled with a coach as *coached students*.

We summarize the class statistics in Figure 1 below. The majority of coached students sign up for the student forum, and many persist with the course to take the final AP exam at the end of the year.

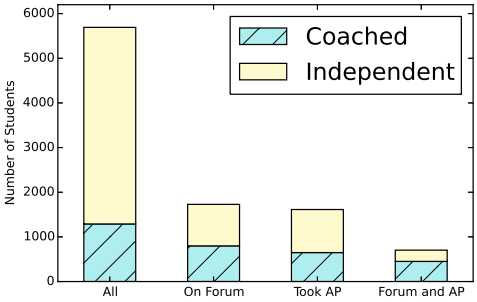


Figure 1: Student participation varies between coached and independent students.

4. EMPIRICALLY CHARACTERIZING SUCCESS OF A HIGH-SCHOOL MOOC

In this section, we use post-test performance and course performance to question the success of MOOCs for high school

students. With an empirical analysis, we provide insights on how to adapt high school MOOCs to benefit different groups of students. To investigate this question, we focus on the subset of students for whom we have post-test data. To evaluate student success in the course, we identify three measures of course participation in MOOCs that are relevant to the high school population: *overall score*, *course completion*, and *post-test score*.

Overall Score The overall score captures the combined score across course assignments, quizzes, exams, and activities, each of which contributes to the final score with some weight. We maintain the same weights as those assigned by the course, exams are weighted most heavily, activities the least.

Overall Score = .3*(Assignment Score + Quiz Score) + .6*Exam Score + .1*Activity Score.

Course Completion The second success measure we use is course completion. Course completion measures the total number of course activities and assessments completed by the student.

$$\textit{Course Completion} = \frac{\text{Total Activities and Assessments Attempted}}{\text{Total Number of Activities and Assessments}}$$

Post-Test Score This score captures student scores in the post test that is conducted 2 weeks after the end of the course. The score ranges from 1 to 5. This score captures the advance placement (AP) score, hence we also refer to it as the AP score.

To evaluate the effectiveness of the high school MOOC on student performance, we first examine the relationship between course completion and course performance. We hypothesize that as students complete a higher percentage of the course, they should do better in the course assessments leading to higher course performance scores and post-test scores. Examining the correlation of course completion to post-test performance, we find that they are positively correlated. This suggests that the course indeed helps students in achieving good performance in the assessments. However, we find that of the students that achieve an overall score of 90 or greater, only 70% pass the post test. Similarly, of the students who complete 90% of the course, only 63% pass the post test. These initial observations indicate the need to perform a more detailed study in order to understand the different student populations in the course.

Next, we examine the relationship between overall score and post-test score, captured in Figure 2. From this plot, we see a positive linear relationship between course performance and post-test score. Notably, we observe that the average post-test score of the students who achieve a 90% or higher in the course is above a 4.0, and well above a passing score.

Students regularly complete three kinds of assessments: assignments, quizzes, and exams. Assignments are programming exercises, testing students' coding abilities. Programming assignments are submitted online through an interface capable of compiling programs and displaying error messages. Quizzes are multiple choice assessments on course material, with an emphasis on recently covered topics. Exams have a similar format to quizzes but are slightly longer. Both quizzes and exams are timed and students cannot change

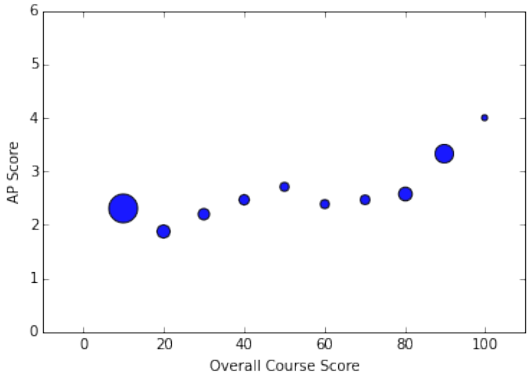


Figure 2: The dot sizes are proportional to the number of students achieving the overall score.

their answers once they submit them. In all, there are 15 assignments, 8 quizzes and 6 exams in the course. We will refer to them as $A_{1:15}$, $Q_{1:8}$, and $E_{1:6}$, in the discussion below.

In Figure 4, we present results of student performance across assessments. Figures 4(a), 4(b), and 4(c) present average student assignment, quiz, and exam scores for students who passed/failed the post test, respectively. We find that students who pass the post test do better on assessments. We also observe that the scores across all assessments show a decreasing trend as the course progresses. This signals that the assessments get harder for both groups of students as the course progresses. Another important observation is the increase in scores for both groups at assignment 8, quiz 5, and exam 4; these assessments are at the start of the second term in the course, indicating that students may have higher motivation at the start of a term.

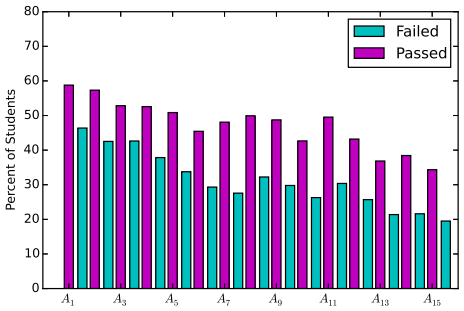
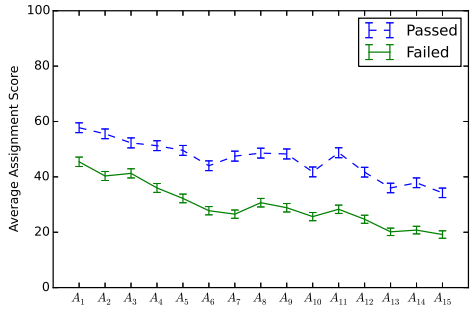


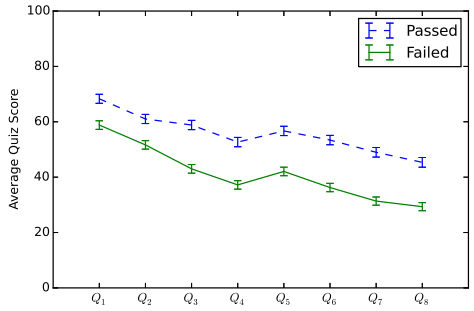
Figure 3: Students who pass are more likely to attempt assignments than students who fail.

Additionally, some assessments show a greater difference between the two groups of students, and performance on these assessments are more informative of student learning. In Figure 4(c), we observe that for both passed and failed students, we see the greatest dip in performance in the final exam. As the final exam is the most comprehensive exam, and possibly most related to the post test, analyzing why students do so poorly on this exam is a worthwhile direction of study in its own right.

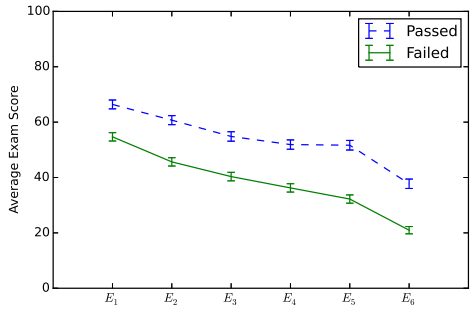
Another important dimension is considering assignment com-



(a) Average assignment scores of passed and failed students



(b) Average quiz scores of passed and failed students



(c) Average exam scores of passed and failed students

Figure 4: Passed students have higher average scores across all assessments than failed students.

pletion rate of these two groups of students. In Figure 3, we examine the relationship between attempting assignments and course performance and find that students passing the post test also attempt more assignments. This implies that the high scores of these students are not only the product of strong prior knowledge, but are also the result of learning from the course.

5. FORUM PARTICIPATION AND POST-TEST PERFORMANCE

In this section, we analyze forum participation of students and examine it's effect on course success. To do so, we answer the following questions:

- Does participation in forums impact post-test performance and learning?
- What are the key differences between participation styles of students who pass the course and students who do not?

We first look at the average score of students who use the forum compared to the average score of students who do not use the forum. Students who use the forum have a statistically higher post test performance score of **2.77**, whereas students who do not use the forum obtain a score of **2.34**, ($p < .001$). It is not clear if the forum impacts learning, or if instead, students with a high desire to learn are more likely to use the forum.

To accurately evaluate forum participation of the two subpopulations, we analyze them on different types of forum participation. Forum participation comprises of different types of student interactions: asking questions, answering other student questions, viewing posts, and contributing to conversation threads. Table 1 gives the comparison of students who pass the post test against student who do not across the various forum participation types. The different types of forum participation types are referred to as: Questions, Answers, Post Views, and Contributions. We also consider the number of days that a student was logged into the forum, which is denoted by Days Online.

On average, students who pass the course make more contributions than students failing in the course. They also answer more questions. Both groups seem to spend roughly the same amount of time online, to view the same number of posts, and to ask the same number of questions. What most distinguishes a student who passes, from one who fails is whether they are answering questions and contributing to conversations.

Forum Behavior	Failed Mean	Passed Mean	Failed Median	Passed Median
Questions	3	4	0	1
Answers	1	4	0	0
Post Views	147	140	73	62
Contributions	9	16	1	2
Days Online	19	21	11	13

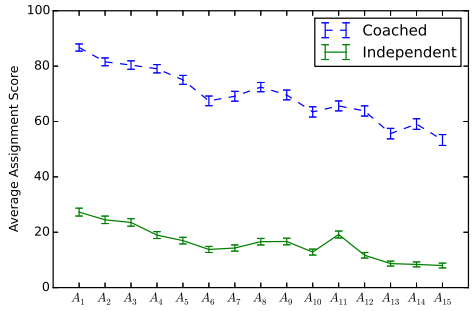
Table 1: The average forum participation is significantly more for students that pass the course. The behavior for which there was a statistical significance difference between the groups are highlighted in bold.

This analysis further demonstrates the importance of forums to MOOCs. Answering questions and contributing to conversations are two behaviors indicative of strong post-test performance. We hope that MOOC designers can use this information to create appropriate intervention and incentive strategies for students.

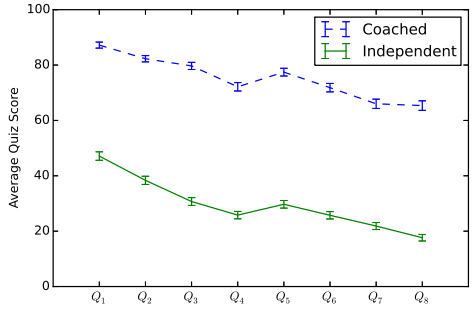
6. COACHING

In this section, we evaluate the effect of coaching on student learning. We compare coached students to independent students using their participation in course assessments and forums. We conclude this section by looking at the subset of students who have only one coach, in order to isolate the effect of coaching from other classroom effects.

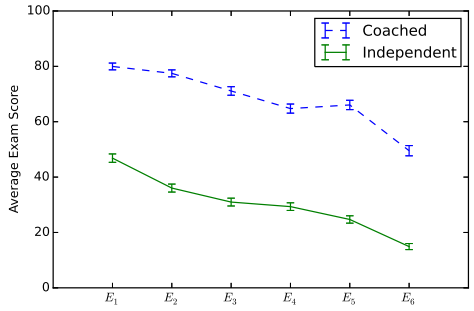
6.1 Course Behavior



(a) Average assignment scores of coached and independent students



(b) Average quiz scores of coached and independent students



(c) Average exam scores of coached and independent students

Figure 5: Coached students have higher average scores than independent students.

We inspect the average assessment scores of coached and independent students in Figure 5. Observing scores across assignments, quizzes, and exams in Figures 5(a), 5(b), and 5(c), respectively, we find that coached students perform better than independent students across all assessments.

Such differentially high performance in the course should indicate higher performance in the AP exam for coached students. However, we see that coached students fail to get a high post-test score. The average post-test score for a coached student is 2.43, while it is 2.59 for an independent student. We test statistical significance using a t-test with a rejection threshold of $p < 0.05$. In Section 6.2, we analyze forum participation of students to understand this difference in scores.

6.2 Forum Participation of Coached and Independent Students

Analyzing forum participation of coached and independent students, we find that there is a significant difference in forum participation between coached and independent students. Table 2 gives the comparison between coached and independent students in forum participation. On average, coached students ask more questions and answer fewer questions on the forums when compared to independent students. Coached students exhibit more passive behavior by predominantly viewing posts rather than writing posts, when compared to independent students. This can be particularly dangerous if the posts which are viewed contain assignment code.

Forum Behavior	Coached Mean	Independent Mean
Questions	2.81	1.90
Answers	1.45	1.72
Post Views	145.49	81.50
Contributions	8.10	7.33
Days Online	20.64	12.55

Table 2: Coached students view more posts and ask more questions. The behavior for which there was a statistical significance difference between the groups are highlighted in bold.

In Table 3, we compare coached students who pass to coached students who fail and see the same differences as those observed between all students who pass, and all students who fail. Students who pass are more likely to answer questions, and contribute to conversations.

Forum Behavior	Passed Mean Coached	Failed Mean Coached
Questions	3.97	2.87
Answers	3.04	0.56
Post Views	141.56	164.14
Contributions	14.19	5.93
Days Online	22.71	21.53

Table 3: The differences in forum behavior between coached students who pass and who fail follow the same trends in forum behavior exhibited by the general population, and shown in Section 5. The behavioral features for which there was a statistical significance difference between the groups are highlighted in bold.

6.3 Coaches with Only One Student

To examine the effect of coaching class size on coached students' post-test performance, we examine coached students in a classroom size of one. Comparing average post test scores of coached students who are singly advised by their coaches (classroom size of one) with independent students, we find that the average post-test score for the coached students is 3.6, while it is 3.2 for independent students. We hypothesize that the lower score of coached students in classroom size greater than one is due to the possibility of sharing answers when students study together. This explains their high overall score but lower post-test scores. This analysis further suggests that the effect of coaching is confounded by the effects of learning in a classroom with peers. To fully

understand the effect of a coach guiding a student through the learning process, the peer-effects of classmates should be better understood and isolated. In Section 7, we take first steps in this direction by proposing student types.

7. INSPECTING UNEXPECTED STUDENT TYPES

In this section, we identify and analyze various types of students in the course based on their performance in the assessments. We classify students into two broad types based on whether the overall scores and post-test scores are correlated. Figure 6 gives the relationship between overall score and post test score for all students. Two groups of students emerge, students who exhibit a correlation between overall scores and post test scores, and students who do not. These two groups can be further broken down based on whether they obtain a high score on the post test, yielding four groups of students.

- *Low learners*: These students have low values for both overall scores and post test scores.
- *High learners*: These student obtain high values for both overall scores and post test scores.
- *Unexpected low learners*: These students obtain high overall scores, but low post test scores.
- *Unexpected high learners*: These students obtain high post test scores, but low overall scores.

Among these, the unexpected low learners and unexpected high learners deviate from the rest of the students. To analyze these two groups, we delve deeper into other aspects of the course such as forum participation and coaching.

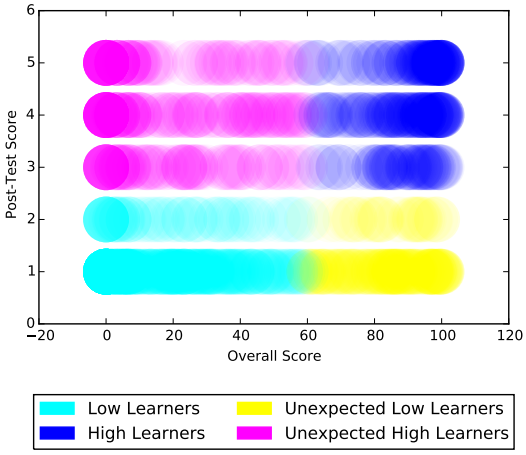


Figure 6: Four groups of students emerge: low learners, high learners, unexpected low and high learners. For high course performance we choose a threshold of 60% as a passing grade.

7.1 Unexpected Low Learners

Unexpected low learners are those students who perform well on the course assessments (with an overall score of over 60%) but who do not earn a passing post-test score. We hypothesize that this might be due to their not retaining information from the course, or not arriving at high overall course scores on their own. To understand their low post-test performance, we examine their forum behavior and coaching environment.

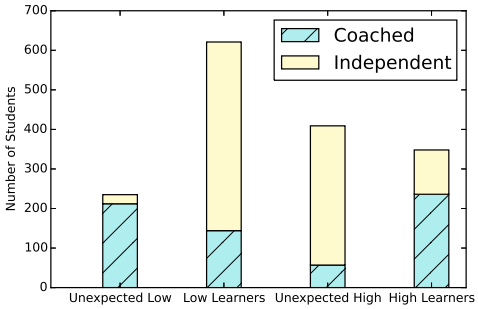


Figure 7: The majority of unexpected low learners are coached, while the majority of unexpected high learners are independent.

As can be seen in Figure 7, approximately 91% of unexpected low learners are coached students. Most of these students are part of large classrooms coached by the same coach, increasing the possibility of getting answers from their peers/coach. Plagiarism is a significant challenge in online courses as proctoring students online is not as efficient as in classroom courses.

Further, analyzing forum performance, we find that approximately 76% of unexpected low learners use the forum. Of those who use the forum, 91% are coached. Table 4 gives the forum participation of coached and independent unexpected low learners. The forum participation of these students have a strong similarity to failing students in Table 1, participating passively in the course by viewing forum posts and contributing to less answers. The coached students are less active than the independent students on the forum in every way, even in post views. While it was posited before that active forum participation is indicative of learning and high AP exam performance, this may not be the case in all groups. For example, the small number of independent students may be using the forum for social, rather than learning purposes.

Forum Behavior	Coached Mean	Independent Mean
Questions	3.5	9.2
Answers	0.5	15.0
Post views	195.0	293.0
Contributions	7.1	67.0
Days Online	25.6	35.2

Table 4: Forum behaviors for which there is a statistical significance between groups are highlighted in bold.

7.2 Unexpected High Learners

Unexpected high performers earn an overall course score of less than 60% but pass the AP exam with a 3 or above. Approximately 86% (357 out of 409) of unexpected high leaners are independent and approximately 80% of the unexpected high learners (323 out of 409) are not on the forums. That this group can do so well on the post test, without either a high amount of course or forum participation strongly suggests that either these students have prior knowledge in computer science or that they are not being primarily exposed to

computer science through this course but are instead using it to supplement another mode of instruction. A pre test of students' prior computer science knowledge would provide further clarity.

8. PREDICTING PERFORMANCE FROM STUDENT BEHAVIOR

In Sections 4 and 5, we see that students' post-test performance is affected by their course and forum behavior. We construct features with which to model these different characteristics of student behavior. These student models are then used to predict post-test scores. By discovering the relative rank of the student model features, we draw insights about student behavior relevant to learning, and to course design.

8.1 Student Model Features

We group the course features from student interactions into four broad categories: 1) course behavior, 2) forum behavior, 3) coaching environment, and 4) topic analysis of forum posts. We extract features from student course behavior and forum behavior, which we describe in Sections 4 and 5. The two other feature categories are described below.

8.1.1 Coaching Environment

Students in the online course are either coached or independent. Coaches are provided a separate discussion forum, apart from the student forum, where they can interact with other coaches and instructors of the course. We extract features that capture coaches' prior knowledge and their involvement in guiding students. Table 5 gives the list of coaching related features extracted from the discussion forum for coaches.

Feature	Explanation
Coached	Boolean feature capturing whether a student is coached or independent
Coach Views	# posts viewed by the coach
Coach Questions	# questions posted by the coach
Coach Answers	# answers posted by the coach
Coach Contributions	# contributions in the forum

Table 5: Coaching related features

8.1.2 Posts Topic Distribution

For extracting topics of the post, we explore the topic modeling framework using Latent Dirichlet Allocation (LDA) [3]. Before using LDA we clean the text data by removing stop words, stemming certain words, and removing all common course words, such as code. To obtain the topic distribution of posts, we use the Machine Learning for Language Toolkit (MALLET) [7]. We use the following parameters for the topic model: number of topics = 150, and optimize-interval = 100, where the hyper-parameters required by LDA, α and β, are set to the default values.

8.2 Predictive Model

We incorporate extracted features in a linear kernel Support Vector Machines (SVM) model, using the python package Scikit-learn [10]. Comparing this model with other machine learning algorithms such as logistic regression, decision trees, and Naive Bayes we found the results to be comparable. We filter our student pool to those who participated in the forums and took the post test (approximately 16% of all students who completed the post test). A subset of features that are predictive of post-test performance were selected using recursive feature elimination in Scikit-learn [10]. Recursive feature elimination works by training a classifier which weighs features and then trims all features with the lowest weights; this trimming allowed us to obtain the best predictions, and to understand which features are most predictive of student success.

8.3 Empirical Results

In this section, we present empirical results using the SVM model defined above to predict post-test performance. To evaluate the effectiveness of this model we compute the F-measure, which is the harmonic mean of precision and recall. F-measure is an optimal metric for a setting with unbalanced classes such as ours, where accuracy may appear to be deceptively high if a classifier reliably predicts the majority class. Our model gives an F-measure of 0.81 for predicting post-test performance. We validate our results with 10-fold cross validation. In the next sections, we analyze the attributes of student behavior which are most predictive of performance.

8.3.1 Topics and Performance

The topics discovered by the topic model fall into four broad categories: help requests, assignments, course material, and course activities. In Table 6, we present the ten topics which are most predictive of post-test performance. The first three topics in the table fall into the help requests category. They include words such as *trouble*, *help*, and *fail*. Four of the top ten topics correspond to assignments, with top words which are descriptive of assignments from the course. For example, in assignment A_4 students are asked to write a program to count the number of hashtags, links, and attributions in a tweet, and in the topic associated with this assignment we see the words: *hashtag*, *tweet*, *attributions*, *mentions*, and *links*. Two topics represent the concepts discussed in the course: object oriented programming, and hash maps. The hash maps topic is particularly interesting as hash maps are not introduced in the course, but students still use them in their projects, and discuss them on the forum. The other prominent topics are topics related to course activities. For example, the activity topic in the the table is an activity given to students to print the location of a vehicle. This is the most elaborate activity that students undertake in the course, hence it appears in the top predictive topics for predicting post-test performance.

Topic Label	Top Words
Help requests	trouble, don't, perfectly, won, updated
Help requests	hope, helps, change, find
Help requests	fail, expected, updated, supposed
Assignment content (A_4)	hashtag, tweet, attributions, mentions, links
Lecture (hashmaps)	Map, key, Getvalue, Hashmap, entry
Course Activity	vehicle, location, backward, forward, GetLocation
Assignment content (A_6)	ArrayList, words, remove, equals, size
Assignment content (A_{10})	strand, size, TurnOn, green, BurntOut
Assignment content (A_{14})	sort, insertion, swap, insert, algorithm
Lecture (OOP and Methods)	object, constructor, methods, parameter, returns

Table 6: Top predictive topics and the words in these topics

Figure 8 gives the distribution of passed and failed students across the different ten most predictive topics given in Table 6. We observe that passing students post about the course activity on vehicles more than failing students. Since activities only contribute to a small portion of their grade,

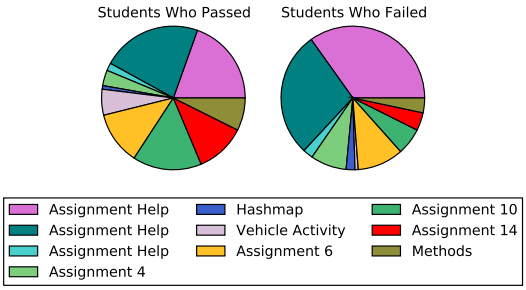


Figure 8: Students who pass post about different topics than students who fail.

participation in activities is a good measure for students' level of motivation and learning.

Additionally, we observe that failing students are far more likely to write posts which fall in the help category. Looking at some of the posts in this category, we find that these posts are often short and use help words, but do not contain detailed information about the specific assignment problem in question. This finding suggests that analyzing the posts for linguistic cues is helpful in understanding students' motivation.

The third important take away from this analysis is that this topic distribution can help discover patterns in student behavior. For example, passing students post about assignment A_{10} more than failing students. But, failing students post more about assignment A_4. As assignments tend to get harder as the course progresses, the difference in behavior can be attributed to failing students needing help on the easier assignments, while the savvier students focus on the harder assignments.

8.3.2 *Critical Assessments*

Here, we describe the most predictive assignments, quizzes and exams that we use in the predictive model. We find that assignments A_4, A_8, A_9, and A_{10} are the most predictive assignments. These assignments are on core concepts and hence may be the most critical assignments in the course. This observation is bolstered by the fact that these assignments are referenced in the forums more than other assignments. Two of these assignments feature in the top ten predictive topics given in Table 6. Pinpointing the moment when a student needs help is not only predictive of their success, but also critical in maintaining engagement and understanding. Understanding which assignments are discussed more in the forums can reveal important information for initiating instructor interventions.

9. CONCLUSION

From this analysis we conclude that MOOCs are a viable option for high school students. Forty-seven percent of students who took the post test passed it. Four hundred and sixty four of these students were to the best of our knowledge self-directed. While we can say that MOOCs work for some high school students, the particularities of this group must be understood. It is not clear, for example, how the students who achieve high course scores, but low AP exam scores are able to do so. Are they receiving answers from other students, or have they truly mastered the course content, but lack the ability to demonstrate this mastery on a test? High school MOOC students are a unique group with particular modeling demands.

We have developed models of these students, characterizing high and low learners by their course and forum behavior, as well as by the topics that they post about. These models have allowed us to differentiate the behavior of students who pass from that of students who fail. In this case study post-test performance was correlated with course-performance, such that students who earned a high course score also earned a high post-test score. Students who performed well on the post test were more likely to contribute to conversations, and to answer questions on the student forum. They were also more likely to post about ungraded activities, and less likely to write posts asking for help. Coached students were more likely to perform well in the course, and spent more time on the forum. Understanding the differences between students who excel and those who do not is crucial in developing the courses that students, and particularly high school students need.

References

[1] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. Engaging with massive online courses. In *International Conference on World Wide Web (WWW)*, 2014.

[2] Ryan Baker, Albert Corbett, Kenneth Koedinger, and Angela Wagner. Off-task behavior in the cognitive tutor classroom: When students "game the system". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2004.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3: 993–1022, 2003.

[4] Gregor Kennedy, Carleton Coffrin, Paula de Barba, and Linda Corrin. Predicting success: How learners' prior knowledge, skills and activities predict MOOC performance. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge (LAK)*, 2015.

[5] René F. Kizilcec, Chris Piech, and Emily Schneider. Deconstructing disengagement: Analyzing learner subpopulations in massive open online courses. In *Conference on Learning Analytics and Knowledge (LAK)*, 2013.

[6] Jaakko Kurhila and Arto Vihavainen. A purposeful MOOC to alleviate insufficient cs education in finnish schools. *Transactions of Computing Education*, 15(2):10:1–10:18, 2015.

[7] Andrew McCallum. MALLET: A machine learning for language toolkit. 2002.

[8] Bock Mike and O'Dea Victoria. Virtual educators critique value of MOOCs for K-12. 2013.

[9] Hedieh Najafi, Rosemary Evans, and Christopher Federico. MOOC integration into secondary school courses. *The International Review of Research in Open and Distributed Learning*, 15(5), 2014.

[10] Fabian Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[11] Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daume III, and Lise Getoor. Learning latent engagement patterns of students in online courses. In *AAAI Conference on Artificial Intelligence*, 2014.

[12] Beth Simon, Julian Parris, and Jaime Spacco. How we teach impacts student learning: Peer instruction vs. lecture in cs0. In *ACM Technical Symposium on Computer Science Education (SIGCSE)*, 2013.

[13] Conrad Tucker, Barton K. Pursel, and Anna Divinsky. Mining student-generated textual data in MOOCs and quantifying their effects on student performance and learning outcomes. In *Conference of American Society for Engineering Education*, 2014.

[14] Lorenzo Vigentini and Andrew Clayphan. Pacing through MOOCs: course design or teaching effect? In *Conference on Educational Data Mining (EDM)*, 2015.

The Affective Impact of Tutor Questions: Predicting Frustration and Engagement

Alexandria K. Vail
Department of Computer Science
North Carolina State University
Raleigh, North Carolina
akvail@ncsu.edu

Joseph B. Wiggins
Department of Computer Science
North Carolina State University
Raleigh, North Carolina
jbwiggi3@ncsu.edu

Joseph F. Grafsgaard
Department of Psychology
North Carolina State University
Raleigh, North Carolina
jfgrafsg@ncsu.edu

Kristy Elizabeth Boyer
Department of Computer & Information Science & Engineering
University of Florida
Gainesville, Florida
keboyer@ufl.edu

Eric N. Wiebe
Department of STEM Education
North Carolina State University
Raleigh, North Carolina
wiebe@ncsu.edu

James C. Lester
Department of Computer Science
North Carolina State University
Raleigh, North Carolina
lester@ncsu.edu

ABSTRACT

Tutorial dialogue is a highly effective way to support student learning. It is widely recognized that tutor dialogue moves can significantly influence learning outcomes, but the ways in which tutor moves, student affective response, and outcomes are related remains an open question. This paper presents an analysis of student affective response, as evidenced by multimodal data streams, immediately following tutor questions. The findings suggest that students' affect immediately following tutor questions is highly predictive of end-of-session self-reported engagement and frustration. Notably, facial action units which have been associated with emotional states such as embarrassment, disgust, and happiness appear to play important roles in students' expressions of frustration and engagement during learning. This line of investigation will aid in the development of a deeper understanding of the relationships between tutorial dialogue and student affect during learning.

Keywords

Tutorial dialogue, affect, frustration, engagement, facial expression

1. INTRODUCTION

Tutorial dialogue provides rich, natural language adaptation to students during learning. An understanding has emerged about the role of interactivity in tutorial dialogue [40, 6] and on dialogue strategies for most effectively supporting students in task-oriented tutorial dialogues [29, 10]. However, a pressing issue is developing an understanding of how specific tutor dialogue moves impact students' affect, and in turn, what influence students' affective responses may have on outcomes.

The need for modeling affect during learning is widely recognized. Research has shown that suites of affect detectors from sensors and log files can perform well but that there are trade-offs depending on the goals of the affect detection modules [22, 33]. Affect detectors have been investigated for a wide variety of affective states including confidence, excitement, frustration, and interest [41], and within tutorial dialogue, for uncertainty [11]. There have also been great strides in sensor-free affect detection which relies primarily on log files [2]. This approach has shown promise during cognitive tutoring [9] and for distinguishing frustration and confusion [27].

Out of all of the affective phenomena that have been examined during learning, two affective states are frustration and engagement. These states have been examined in fine-grained analyses as tutoring unfolds, and also as outcome measures regarding students' perceptions of the success of the tutoring session. Engagement and frustration have been predicted at above-chance levels using facial expression-based affect detection even without the presence of interactive events during text or diagram comprehension [5]. Engagement and frustration have also been predicted with nonverbal behaviors, including facial expression, after student task events during problem solving [16]. In a compelling development, emerging evidence shows that fine-grained affective events can have long-lasting relationships with outcomes that may be far removed from those affective events [36].

This paper advances the understanding of student emotions in learning by examining students' fine-grained affective responses to tutor questions during tutorial dialogue. It investigates the hypothesis that students' affective responses immediately following tutor questions are related to self-reported frustration and engagement at the end of the session. The results indicate that several key facial expression

DOI: 10.1145/1235

features immediately following two different types of tutor questions are highly predictive of end-of-session self-reported engagement and frustration. This line of investigation represents a step forward in understanding the affective impact of tutorial strategies.

2. RELATED WORK

Tutorial dialogue researchers have long studied what human tutors naturally do: how strategies differ between experts and novice tutors [12] whether Socratic or didactic approaches are most effective [35] and how tutors scaffold and fade support during problem solving [4], among others. The impact of particular tutorial dialogue moves has been the focus of significant attention, with findings indicating that positive and negative feedback have different impact based on students' self-efficacy level [3], that bottom-out directives are not conducive to learning [29], and that adapting to student uncertainty improves the effectiveness of tutorial dialogue [10]. However, this paper examines a different aspect of these tutorial dialogue moves that is critical in learning: students' affective response as expressed on the face and as embodied in gestures.

Multimodal features such as dialogue, facial expression, posture, and task actions have been used to predict affective states, such as boredom, confusion, excitement, and frustration, as those states occur during learning [23, 8, 7]. Moreover, multimodal features such as facial expression and gestures can significantly predict frustration and engagement reported at the end of tutoring sessions [17], and some differences have emerged in the extent to which upper and lower facial expression features are associated with these outcomes [15]. This previous work on utilizing multimodal features for predicting frustration and engagement during human-human tutoring has emphasized the important role that tutor dialogue moves play in affective outcomes. Other factors, such as student personality profile, can also contribute significantly to predicting these outcomes [39]. The present work examines moment-by-moment affect as evidenced by multimodal traces, and then analyzes the relationship between these multimodal behaviors and the outcomes of frustration and engagement as reported by students after the tutoring session.

3. STUDY DATA

The present analysis investigates the multimodal behavior of students during a computer-mediated tutorial session in introductory computer science, and specifically in Java programming [18, 30]. The tutorial interface, shown in Figure 1, is divided into four panes: the task description, the student's Java source code, the compilation and execution output of the program, and the textual dialogue messages between the tutor and the student. The tutor's interactions with the environment were constrained to progression between tasks and sending textual messages to the student.

Students ($N = 67$) were university students in the United States enrolled in an introductory engineering course, with an average age of 18.5 years ($s = 1.5$ years), whereas the human tutors ($N = 5$) were primarily graduate students with previous experience in tutoring or teaching introductory programming. The behavior of the student was collected using a set of multimodal sensors, as shown in Figure 2, including a Kinect depth sensor, an integrated webcam, and a skin conductance bracelet. The following subsections detail the modalities appearing significant in the present analysis.

Each student participated in six 40-minute sessions over the course of four weeks; however, the present analysis only examines data from the first lesson. Before and after each lesson, students completed a content-based pretest and identical posttest; the tutoring sessions were found to be significantly effective in facilitating learning gains ($p \ll 0.0001$). In addition to the posttest, students also completed a post-survey, including the NASA-TLX workload survey [20] and the User Engagement Survey [32]. The present analysis investigates self-reported *frustration*, taken from the Frustration Level item of the NASA-TLX workload survey, and *engagement*, taken as an average of three sub-scales of the User Engagement Survey: Focused Attention (perception of time passing), Felt Involvement (perception of involvement with the session), and Endurability (perception of the activity as worthwhile).

3.1 Task Event and Dialogue Features

During the tutoring session, the interface described above logged tutor and student dialogue messages, student typing in the code window, and student progress through the task. No turn-taking measures were enforced in the dialogue: students and tutors could send messages to the other at any point. All exchanged messages were automatically tagged by a J48 decision tree classifier [37] with a dialogue act annotation scheme created for task-oriented tutorial dialogue that differentiates tutor questions, feedback, and hints, among other dialogue moves [38]. In that work, the Cohen's kappa between two human annotators was 0.87 and the Cohen's kappa between human and the J48 decision tree classifier was 0.786.

The analysis presented here focuses on two types of tutor dialogue moves: inference questions and evaluative questions. (Although other question types were investigated, student reactions to these were not found to have significant predictive power.) Inference questions require the formation of an action plan or reasoning about existing content knowledge. For example, *'How do you think this problem can be solved?'*, or *'How can you fix this error?'* are considered to be inference questions. On the other hand, evaluative questions aim to evaluate the student's belief in his or her own understanding of the material, e.g., *'Does that make sense so far?'*, or *'Do you understand?'* (see Figure 4).

Previous work has suggested that questions can stimulate cognitive disequilibrium in a student [34], which is often considered to be a critical step in knowledge acquisition [13]. On the other hand, evaluative questions that ask a novice to evaluate whether she understands material may not be particularly helpful pedagogically because novices often cannot identify what they do not understand, or may be hesitant to speak up even if they are aware that they are confused. Nonetheless these questions occurred regularly in our corpus with experienced (though not expert) human tutors. We investigate whether students' affective response to these types of tutor dialogue moves is significantly predictive of student engagement and frustration as reported at the end of the session.

3.2 Facial Expression Features

Student facial expressions were automatically extracted

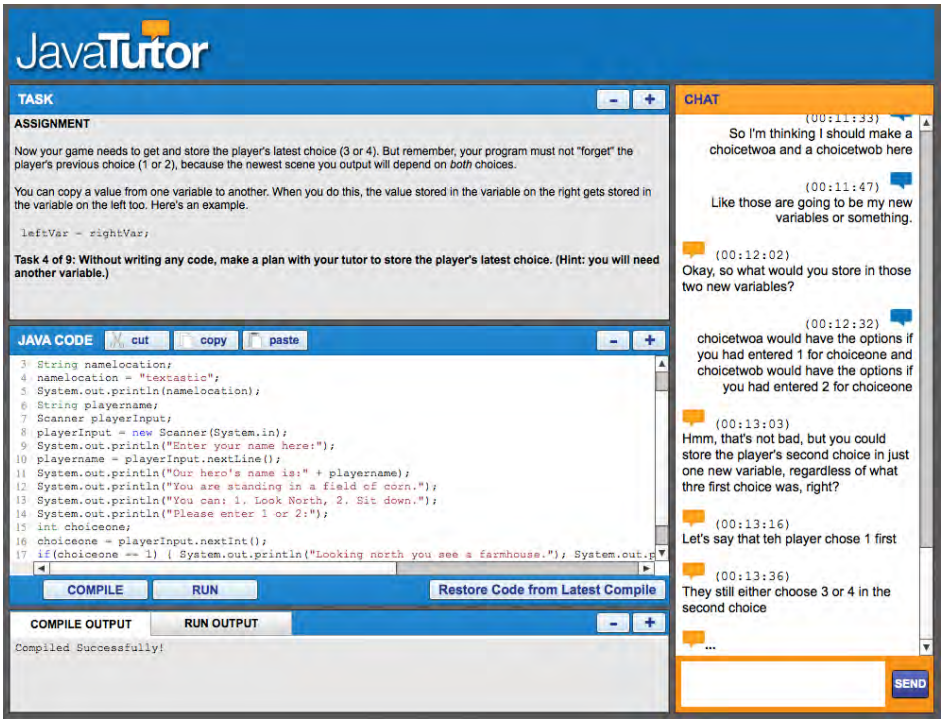


Figure 1: The web-based tutorial interface for Java programming.

using a state-of-the-art facial expression recognition toolbox, FACET (commercial software preceded by a research version known as the Computer Emotion Recognition Toolbox, CERT) [26]. FACET tracks the frame-by-frame presence of several facial action units according to the Facial Action Coding Scheme [25]. These action units include movements such as AU6 Cheek Raiser, AU12 Lip Corner Puller, AU24 Lip Pressor, and AU26 Jaw Drop (see Figures 5 and 6 for illustration). For each facial action unit, the FACET software suggests an *Evidence* measure, indicating the chance that the target expression is present. This *Evidence* measure is on a scale where negative values represent evidence of the absence of a facial expression and positive values indicate evidence of the presence of one. The more positive the measure, the more confident FACET is that the feature is present.

3.3 Gesture Features

The Kinect depth camera also tracked hand-to-face gestures made by the student during the tutoring session. An algorithm developed to detect such gestures was developed to recognize one or two hands touching the lower face. In order to do this, the algorithm relies on surface propagation from the center of the head, identifying round (i.e. a normal head shape) or oblong shapes (i.e., shapes extending beyond the normal head shape) based on distances from the center of the head. This gesture detection algorithm was previously found to be 92.6% accurate when compared against manual labels [14].

4. ANALYSIS

The present analysis focuses on the affective response of a student, as observed by multimodal traces of face and gesture, after tutor inference questions and evaluative questions. We hypothesize that multimodal features after these tutor questions can predict student engagement and frustration. In particular, we examine three seconds after each tutor dialogue move (a manually-determined interval). The multimodal response of the student was characterized using the following categories of features, all of which were provided to the predictive models. However, note that only the first two of these categories of features (shown in bold below) appear significantly predictive within the models.

1. **Average evidence measure for each of the facial expression action units during the interval (19 features)**
2. **Percentage of the interval in which a one-hand-to-face or two-hands-to-face gesture was observed (2 features)**
3. Number of skin conductance responses identified during the interval as measured by a skin conductance response bracelet (1 feature)
4. Average student distance from the workstation during the interval (1 feature)
5. Average difference between the highest and lowest points of the student's body from the workstation during the interval, indicating leaning (1 feature)

We calculated the average value of each multimodal feature listed in the categories above across each tutoring session. For each feature, we computed its conditional probability of occurring after the tutor moves of inference question or evaluative question. We also provided the model with the overall occurrence of that feature across the entire tutoring

Figure 2: Multimodal instrumented tutoring session, including a Kinect depth camera to detect posture and gesture, a webcam to detect facial expression changes, and a skin conductance bracelet to detect electrodermal activity.

Figure 3: Dialogue excerpt illustrating a tutor inference question in context.

	Student compiles the program, encounters an error.
STUDENT	Oh.
TUTOR	So how can we fix this?
STUDENT	Hmm.
STUDENT	Switch the prompt line with the response line?
TUTOR	Okay, try it.

Figure 4: Dialogue excerpt illustrating a tutor evaluative question in context.

STUDENT	Do I need to set the player input before line 13?
TUTOR	The `while` tests that [variable]. You need to be sure it enters the loop at least once.
TUTOR	Good.
TUTOR	Does that make sense?
STUDENT	Yeah.
STUDENT	But what happens if I don't enter 1 or 2?

session in order to control for the influence of the feature overall (rather than only after the tutor moves of interest). Specifically, the features conditional on tutor moves were averages of the form $Avg(Feature|TutorQ)$ for each student that completed the session. The session-wide average of each feature, $Avg(Feature)$ were also provided to the model for each multimodal feature in all of the categories above.

Standardization was performed on each feature by subtracting the mean and dividing by the standard deviation, so that the regression coefficients would be more interpretable. The standardized features were provided to a stepwise regression modeling procedure optimizing for the leave-one-student-out cross-validated R^2 value (the coefficient of determination), while at the same time requiring a strict $p < 0.05$ cut-off value after Bonferroni correction on significance values.

5. RESULTS AND DISCUSSION

For both types of tutor question, evaluative and inference, a predictive model was built to predict student frustration and student engagement, resulting in a potential four models. Three of the four models uncovered significant predictive relationships. The following subsections detail models predicting frustration after tutor inference and evaluative questions, and a model predicting engagement after tutor evaluative questions.

5.1 Frustration

The results suggest that student facial expressions are significantly predictive of self-reported end-of-session frustration. The predictive model for student frustration based on tutor evaluative questions includes two features, both of which are facial action units occurring in the three-second interval following the tutor evaluative question (Table 1).

Two facial action unit features after tutor evaluative ques-

[1]The models reported in this paper were built as a part of a larger exploratory analysis. As a result, the p-values reported have been modified by a Bonferroni correction

Table 1: Predictive model for standardized end-of-session frustration after tutor evaluative questions (TutorQE).[1]

Frustration =	R^2	p
−0.7039 * AU12 after TutorQE	0.0764	0.014
−0.6279 * AU28 after TutorQE	0.2471	0.030
−0.1635 (Intercept)		1.000
Leave-One-Out Cross-Validated $R^2 = 0.3235$		

Table 2: Predictive model for standardized end-of-session frustration after tutor inference questions.[1]

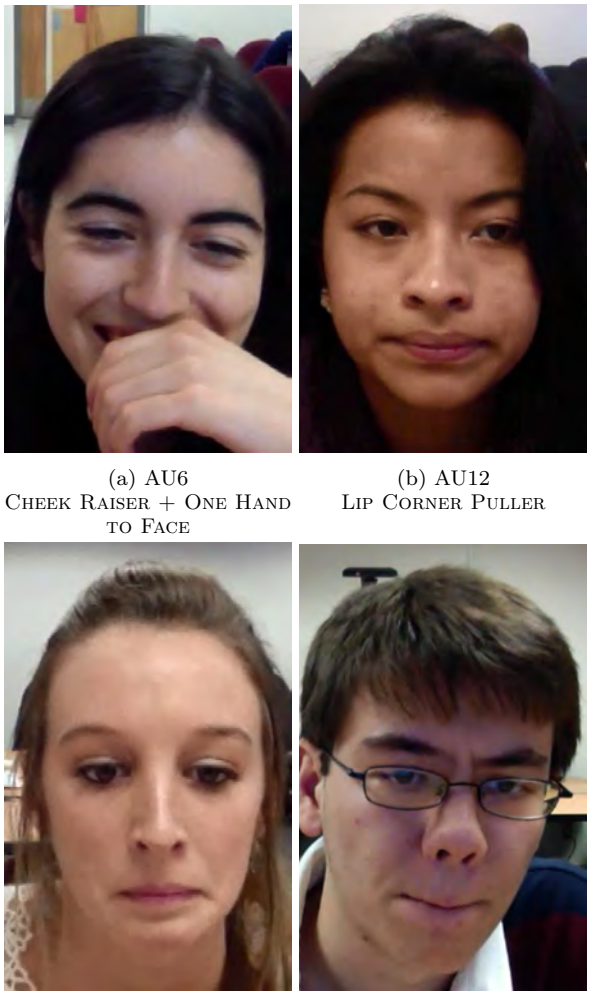
Frustration =	R^2	p
+0.5660 * AU6 after TutorIQ	0.2893	0.022
+0.3635 * AU20	0.0499	0.019
−0.0174 (Intercept)		1.000
Leave-One-Out Cross-Validated $R^2 = 0.3392$		

tions are significantly predictive of student frustration. Higher intensity levels of of AU12 Lip Corner Puller (Figure 5b) following a tutor evaluative question are *negatively* indicative of frustration, as is the presence of AU28 Lip Suck (Figure 5d). AU12 is associated with smiling, which is typically not associated with frustration although on occasion, the two can go hand in hand [21].

AU 28 is a type of lower face movement sometimes associated with fidgeting, and this type of motion may be a "self-manipulator" that is part of emotion regulation. It is possible that students engaged in this challenging learning task may exhibit this movement to alleviate negative emotions related to frustration, resulting in lower self-reported frustration at the end of the session. When students are faced with a question that asks them to evaluate whether they understand the material being tutored, these facial expressions may both reflect the presence of emotion regulation that could mitigate the students' overall feeling of frustration.

The next model examines student responses to tutor inference questions. In contrast to evaluative questions, inference questions ask students to bring pieces of knowledge together to infer the answer to a question and then to express a substantive answer. Two facial action unit features exhibited following these questions appear as significantly predictive of student frustration. The model shows that AU6 Cheek Raiser (Figure 5a) after tutor inference questions is positively predictive of frustration, as is the overall session occurrence of AU20 Lip Stretcher (Figure 5c). The model is displayed in Table 2.

Interestingly, AU6 has been related to pain expressions in the literature on pain detection [28]. When asked to answer an inference question, it is possible that students exhibited a "pained" expression that coincides with frustration. The expression of AU20 has been observed to coincide with moments of embarrassment or awkwardness [24], when people were embarrassed or amused in the period after doing directed facial actions (the technique used to develop images for the Facial Action Coding System). AU20 only occurred among embarrassed participants in that study. When faced with a tutor inference question, this expression may indicate that the student is unsure, awkward, or embarrassed, which may unsurprisingly be related to frustration. Deeper future investigation of subsequent student dialogue moves will help elucidate this phenomenon.



(a) AU6 Cheek Raiser + One Hand to Face

(b) AU12 Lip Corner Puller

(c) AU20 Lip Stretcher

(d) AU28 Lip Suck

Figure 5: Sample frames from the student webcam illustrating the facial action unit features appearing in the predictive models for student frustration, as identified by FACET.

5.2 Engagement

Next we built models to predict student engagement based on affective responses to tutor inference questions and evaluative questions. For inference questions, none of the features provided to the model were predictive of engagement. However, for affective response to tutor evaluative questions, there were seven predictive features, three of which are specific to the interval following the event, and four of which are session-wide (Table 3).

The model suggests that facial expression features account for most of the variance in predicting student engagement; however, one session-wide gesture feature was also

[1] $p \leq \alpha/n$, where $n = 21$ is the number of statistical tests conducted in the larger analysis, in order to reduce the familywise error rate to $\alpha = 0.05$.

Table 3: Predictive model for standardized engagement after tutor evaluative questions.[1]

Engagement =	R^2	p
+0.4422 * OneHTF	0.1815	< 0.001
−0.5989 * AU10 after TutorEQ	0.1831	< 0.001
+0.5770 * AU12	0.2280	< 0.001
+0.5097 * AU26 after TutorEQ	0.0514	< 0.001
−0.2941 * AU2	0.1923	0.003
+0.2467 * AU5	0.0295	0.002
+0.1792 * AU24 after TutorEQ	0.0566	0.018
+0.4100 (Intercept)		1.000
Leave-One-Out Cross-Validated $R^2 = 0.9224$		

selected. The more frequently a student was displaying a OneHandToFace gesture, which may indicate thoughtful contemplation, the more engaging the student reported the experience at the end of the session.

Three more session-wide facial expression features were selected as significantly predictive of student engagement. The more intense the expression of AU12 Lip Corner Puller (Figure 5b) or AU5 Upper Lid Raiser (Figure 6b), the more engaged the student. For AU12 which is often associated with smiling, a positive emotion is likely related to higher engagement. In this task, AU5 is likely associated with the student looking at the screen, possibly indicating paying attention and focusing on the task (as opposed to the opposite facial movement of blinking or shutting one's eyes). In contrast, AU2 Outer Brow Raiser (Figure 6a) was predictive of lower engagement. This action unit is a component of the "fear brow" (AU1+2+4) which has been evidenced as a display of anxiety [19].

Narrowing down to the context of three seconds after tutor evaluative questions, three facial expression features were significantly correlated with student engagement. The more that a student expresses AU26 Jaw Drop (Figure 6e), or the more that the student expresses AU24 Lip Pressor (Figure 6d), the more engaged the student reported being at the end of the session. Jaw drop is a dynamic action unit that may occur when the mouth is closed or already partly open. In either case, this action unit may be associated with focus on the task, although it could also plausibly be associated with a yawn (which we would not expect to coincide with higher engagement). With respect to AU24, which is a prototypical component of anger, an important interplay of learning and affect expression emerges. Some facial movements that are part of prototypical displays of negative basic emotions, such as anger, appear to be indicative of mental effort during learning, rather than negative affect [31]. From this perspective, it makes sense that this AU24 would be related to engagement. On the other hand, the more that a student expressed AU10 Upper Lip Raiser (Figure 6c) during this interval, the *less* engagement reported by the student at the end of the session. This action unit, which is a component of prototypical disgust, is likely to run contrary to engagement.

6. CONCLUSION

Tutor dialogue moves in one-on-one human tutoring significantly influence student outcomes, both cognitive and affective. This paper has examined students' affective response to two types of tutor questions: inference questions which require some reasoning to construct an answer, and evaluative questions, which ask students to reflect on the extent to which they understand the material. The results show that immediately after these tutor questions, students' affective displays—particularly with respect to facial expression—are highly predictive of the outcomes of frustration and engagement. By detecting these affective displays which have been associated in prior studies with emotions such as embarrassment, disgust, or happiness, we can begin to understand the moment-by-moment affective processes that influence learning through tutorial dialogue, and relate those fine-grained events to overall outcomes.

While these facial movements have been associated with prototypical emotion displays in the literature, it is important to further contextualize the moments in which these expressions appear during tutoring. For instance, action units typically associated with anger are likely indicators of mental effort during learning. Similarly, an action unit associated with disgust (e.g., AU10) may be related to students' appraisal of the tutor's question in the moment. Further research seek to ground these interpretations more extensively across salient moments of tutoring.

There are several additional directions for future work. Detecting important moments during tutoring is an open area of investigation, with evidence suggesting that moment-by-moment affect may be related to distal outcomes [36, 1]. In future work, it will be important to expand our understanding of the identified non-verbal predictors for frustration and engagement more deeply. We must consider a wider variety of contexts, and explore different widths of time after tutorial events to examine affective responses with longer (or shorter) times to manifest. It is hoped that this line of investigation will lead to richer affect models for tutorial dialogue.

Acknowledgments

The authors wish to thank the members of the LearnDialogue and Intellimedia groups at North Carolina State University for their helpful input. This work is supported in part by the Department of Computer Science at North Carolina State University and the National Science Foundation through Grants IIS-1409639, CNS-1453520, and a Graduate Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the participants, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

7. REFERENCES

[1] R. S. Baker, A. B. Goldstein, and N. T. Heffernan. Detecting learning moment-by-moment. *International Journal of Artificial Intelligence in Education*, 21(1-2):5–25, 2011.

[2] R. S. Baker, S. M. Gowda, M. Wixon, J. Kalka, A. Z. Wagner, A. Salvi, V. Aleven, G. W. Kusbit, J. Ocumpaugh, and L. Rossi. Towards sensor-free affect detection in cognitive tutor algebra. In *Proceedings of the 5th International Conference on Educational Data Mining (EDM 2012)*, pages 126–133, 2012.

[3] K. E. Boyer, R. Phillips, M. Wallis, M. Vouk, and J. Lester. Balancing cognitive and motivational scaffolding in tutorial dialogue. In *Intelligent Tutoring Systems*, pages 239–249. Springer, 2008.

[4] W. L. Cade, J. L. Copeland, N. K. Person, and S. K. D'Mello. Dialogue modes in expert tutoring. In *Intelligent tutoring systems*, pages 470–479. Springer, 2008.

[5] Y. Chen, N. Bosch, and S. D'Mello. Video-based affect detection in noninteractive learning environments.

[6] M. T. Chi, S. A. Siler, H. Jeong, T. Yamauchi, and R. G. Hausmann. Learning from human tutoring. *Cognitive Science*, 25(4):471–533, 2001.

[7] D. G. Cooper, K. Muldner, I. Arroyo, B. P. Woolf, and W. Burleson. Ranking feature sets for emotion models used in classroom based intelligent tutoring systems. In *User Modeling, Adaptation, and Personalization*, pages 135–146. Springer, 2010.

[8] S. K. D'Mello and A. Graesser. Multimodal semi-automated affect detection from conversational cues, gross body language, and facial features. *User Modeling and User-Adapted Interaction*, 20(2):147–187, 2010.

[9] S. E. Fancsali. Causal discovery with models: behavior, affect, and learning in cognitive tutor algebra. In *Proceedings of the 7th International Conference on Educational Data Mining (EDM 2014)*, pages 28–35. Citeseer, 2014.

[10] K. Forbes-Riley and D. Litman. Benefits and challenges of real-time uncertainty detection and adaptation in a spoken dialogue computer tutor. *Speech Communication*, 53(9):1115–1136, 2011.

[11] K. Forbes-Riley and D. J. Litman. Adapting to student uncertainty improves tutoring dialogues. In *AIED*, pages 33–40, 2009.

[12] M. Glass, J. H. Kim, M. W. Evens, J. A. Michael, and A. A. Rovick. Novice vs. expert tutors: A comparison of style. In *MAICS-99, Proceedings of the Tenth Midwest AI and Cognitive Science Conference*, pages 43–49, 1999.

[13] A. C. Graesser and B. A. Olde. How does one know whether a person understands a device? The quality of the questions the person asks when the device breaks down. *Journal of Educational Psychology*, pages 524–536, 2003.

[14] J. F. Grafsgaard, R. M. Fulton, K. E. Boyer, E. N. Wiebe, and J. C. Lester. Multimodal Analysis of the Implicit Affective Channel in Computer-Mediated Textual Communication. In *Proceedings of the 14th International Conference on Multimodal Interaction*, pages 145–152, 2012.

[15] J. F. Grafsgaard, J. B. Wiggins, K. E. Boyer, E. N. Wiebe, and J. C. Lester. Automatically recognizing facial expression: Predicting engagement and frustration. In *Proceedings of the 6th International Conference on Educational Data Mining (EDM 2013)*, pages 43–50, 2013.

[16] J. F. Grafsgaard, J. B. Wiggins, K. E. Boyer, E. N. Wiebe, and J. C. Lester. Predicting learning and affect from multimodal data streams in task-oriented tutorial dialogue. In *Proceedings of the 7th International Conference on Educational Data Mining (EDM 2014)*, pages 122–129, 2014.

[17] J. F. Grafsgaard, J. B. Wiggins, A. K. Vail, K. E. Boyer, E. N. Wiebe, and J. C. Lester. The additive value of multimodal features for predicting engagement, frustration, and learning during tutoring. In *Proceedings of the 16th International Conference on Multimodal Interaction*, pages 42–49. ACM, 2014.

[18] E. Y. Ha, J. F. Grafsgaard, C. M. Mitchell, K. E. Boyer, and J. C. Lester. Combining Verbal and Nonverbal Features to Overcome the 'Information Gap' in Task-Oriented Dialogue. In *Proceedings of the 13th Annual SIGDIAL Meeting on Discourse and Dialogue*, pages 247–256, 2012.

[19] J. A. Harrigan and D. M. O'Connell. How do you look when feeling anxious? facial displays of anxiety. *Personality and Individual Differences*, 21(2):205–212, 1996.

[20] S. G. Hart and L. E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Advances in Psychology*, pages 139–183, 1988.

[21] M. E. Hoque, D. J. McDuff, and R. W. Picard. Exploring temporal patterns in classifying frustrated and delighted smiles. *Affective Computing, IEEE Transactions on*, 3(3):323–334, 2012.

[22] S. Kai, L. Paquette, R. S. Baker, N. Bosch, S. D'Mello, J. Ocumpaugh, V. Shute, and M. Ventura. A comparison of video-based and interaction-based affect detectors in physics playground.

[23] A. Kapoor and R. W. Picard. Multimodal affect recognition in learning environments. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 677–682. ACM, 2005.

[24] D. Keltner. Signs of appeasement: Evidence for the distinct displays of embarrassment, amusement, and shame. *Journal of Personality and Social Psychology*, 68(3):441–454, 1995.

[25] M. R. Lepper and M. Woolverton. The wisdom of practice: Lessons learned from the study of highly effective tutors. *Improving Academic Achievement: Impact of Psychological Factors on Education*, pages 135–158, 2002.

[26] G. Littlewort, J. Whitehill, T. Wu, I. Fasel, M. Frank, M. Javier, and M. Bartlett. The computer expression recognition toolbox (CERT). In *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition and Workshops*, pages 298–305, 2011.

[27] Z. Liu, V. Pataranutaporn, J. Ocumpaugh, and R. S. Baker. Sequences of frustration and confusion, and learning. In *Proceedings of the 6th International Conference on Educational Data Mining (EDM 2013)*, pages 114–120, 2013.

[28] P. Lucey, J. F. Cohn, I. Matthews, S. Lucey, S. Sridharan, J. Howlett, and K. M. Prkachin. Automatically detecting pain in video through facial action units. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(3):664–674, 2011.

[29] C. M. Mitchell, E. Y. Ha, K. E. Boyer, and J. C. Lester. Learner characteristics and dialogue: recognising effective and student-adaptive tutorial

strategies. *International Journal of Learning Technology 25*, 8(4):382–403, 2013.

[30] C. M. Mitchell, E. Y. Ha, K. E. Boyer, and J. C. Lester. Learner characteristics and dialogue: recognising effective and student-adaptive tutorial strategies. *International Journal of Learning Technology*, 8(4):382–403, 2013.

[31] M. Mortillaro, M. Mehu, and K. R. Scherer. Subtly Different Positive Emotions Can Be Distinguished by Their Facial Expressions. *Social Psychology and Personality Science*, 2(3):262–271, 2011.

[32] H. L. O'Brien and E. G. Toms. The development and evaluation of a survey to measure user engagement. 61(1):50–69, 2010.

[33] L. Paquette, B. Mott, K. Brawner, J. Rowe, J. Lester, R. Sottilare, R. Baker, J. DeFalco, and V. Georgoulas. Sensor-free or sensor-full: A comparison of data modalities in multi-channel affect detection. In *Under review for the 8th International Conference on Educational Data Mining,(under review)*.

[34] J. Piaget. *The origins of intelligence*. International University Press, 1952.

[35] C. P. Rosé, J. D. Moore, K. VanLehn, and D. Allbritton. A comparative evaluation of socratic versus didactic tutoring. *Proceedings of Cognitive Sciences Society*, pages 869–874, 2001.

[36] M. O. Z. San Pedro, E. L. Snow, R. S. Baker, D. S. McNamara, and N. T. Heffernan. Exploring dynamical assessments of affect, behavior, and cognition and math state test achievement.

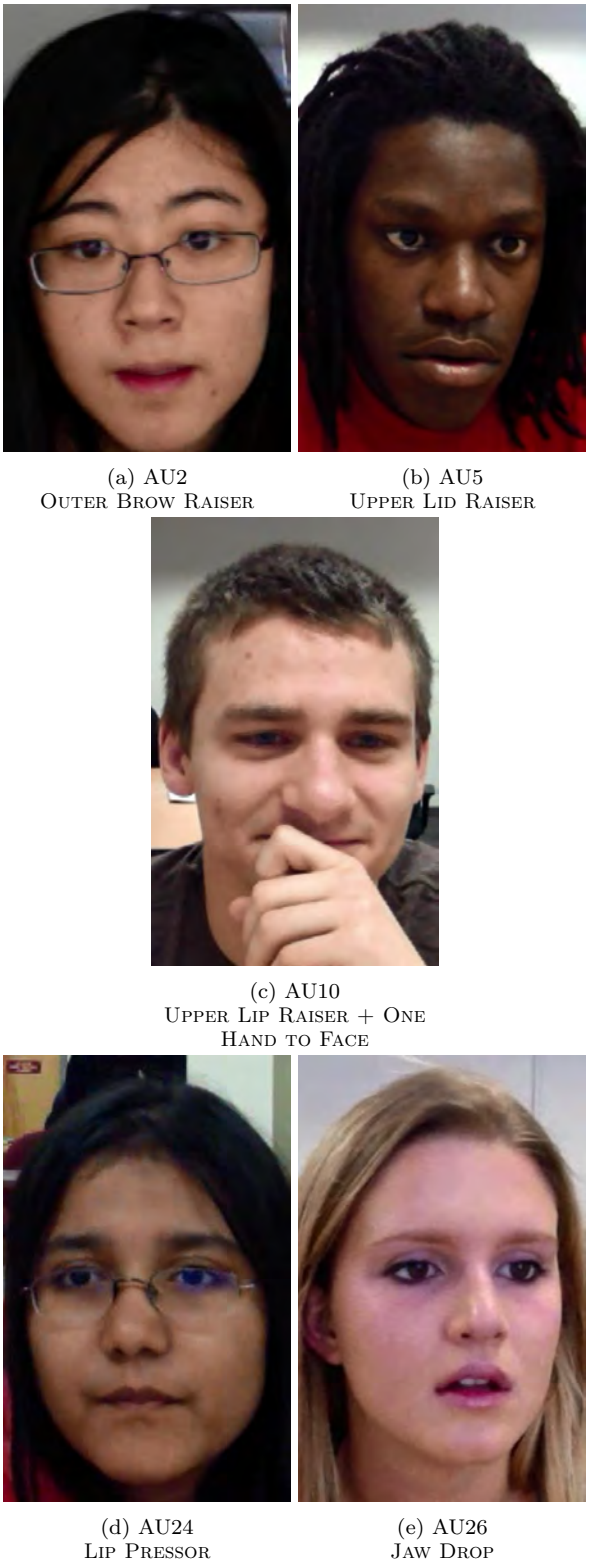
[37] A. K. Vail and K. E. Boyer. Adapting to Personality Over Time: Examining the Effectiveness of Dialogue Policy Progressions in Task-Oriented Interaction. In *Proceedings of the 15th Annual SIGDIAL Meeting on Discourse and Dialogue*, pages 41–50, 2014.

[38] A. K. Vail and K. E. Boyer. Identifying Effective Moves in Tutorial Dialogue: On the Refinement of Speech Act Annotation Schemes. In *Proceedings of the 12th International Conference on Intelligent Tutoring Systems*, pages 199–209, 2014.

[39] A. K. Vail, J. F. Grafsgaard, J. B. Wiggins, J. C. Lester, and K. E. Boyer. Predicting learning and engagement in tutorial dialogue: A personality-based model. In *Proceedings of the 16th International Conference on Multimodal Interaction*, pages 255–262. ACM, 2014.

[40] K. VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

[41] M. Wixon, I. Arroyo, K. Muldner, W. Burleson, C. Lozano, and B. Woolf. The opportunities and limitations of scaling up sensor-free affect detection. pages 145–152, 2014.



(a) AU2 Outer Brow Raiser

(b) AU5 Upper Lid Raiser

(c) AU10 Upper Lip Raiser + One Hand to Face

(d) AU24 Lip Pressor

(e) AU26 Jaw Drop

Figure 6: Sample frames from the student webcam illustrating the facial action unit features appearing in the predictive model for student engagement, as identified by FACET. Note that AU12 Lip Corner Puller (Figure 5b) also appears in these models.

Unnatural Feature Engineering: Evolving Augmented Graph Grammars for Argument Diagrams

Linting Xue
North Carolina State University
Raleigh, North Carolina, USA
lxue3@ncsu.edu

Collin F. Lynch
North Carolina State University
Raleigh, North Carolina, USA
cflynch@ncsu.edu

Min Chi
North Carolina State University
Raleigh, North Carolina, USA
mchi@ncsu.edu

ABSTRACT

Graph data such as argument diagrams has become increasingly common in EDM. Augmented Graph Grammars are a robust rule formalism for graphs. Prior research has shown that hand-authored graph grammars can be used to automatically grade student-produced argument diagrams. But hand-authored rules can be time consuming and expensive to produce, and they may not generalize well to novel contexts. We applied Evolutionary Computation to automatically induce empirically-valid graph grammars for argument diagrams that can be used for automatic grading or provide the basis for hints. Our results show that our approach can generate more relevant rules than experts or other state of the art algorithms, and that these evolved rules outperform the alternatives.

Keywords

Evolutionary Computation, Augmented Graph Grammars, Argument Diagramming, Feature Engineering

1. INTRODUCTION

Intelligent tutoring systems and computer-supported collaboration platforms have grown increasingly popular in recent years. As they have grown in popularity they have also been applied in increasingly complex domains such as argumentation [14], legal reasoning [22] and writing [6]. MOOCs and other online educational platforms have also grown in popularity yielding large repositories of user-system interaction logs [10], and classical tutors and educational games have grown more common in classrooms yielding large repositories of student data [13]. Much of this data can be represented as rich graph structures such as argument diagrams [17] or interaction networks [7].

Despite the increasing prevalence of graph data, comparatively little work has been done on automatically evaluating student-produced graphs or graph logs. In prior work we demonstrated that hand-authored Graph Grammars can be used as features to automatically grade student-produced argument diagrams [16, 17]. But hand-authoring complex rules is time consuming, expensive, and does not generalize well to novel contexts. Other authors have developed analytical tools tuned to path analysis [24, 3], however these are tailored to a specific task. Other more general purpose algorithms (e.g. [30, 5]) have limitations and are unsuited to the induction of generalized rules that use negation or other complex elements. Therefore it has not yet been shown that it is possible to *automatically* induce complex, empirically-valid, rules for rich graph structures that are comparable to rules produced by domain experts.

In this paper we will describe our work on the automatic induction of Augmented Graph Grammars for student-produced argument diagrams. Our goal in this work is to explore ways to automatically induce empirically-valid graph rules that can be used as *features* for automatic grading and which can provide the basis for hints. While our previous work was focused on inducing positive rules in [33] and in [19], in this work we applied Evolutionary Computation (EC) to induce both positive and negative rules for student graphs that incorporate more complex elements such as negation and generalized types. Additionally, in our previous work we compared the induced rules with a small number of expert rules while in this work, we will compare our induced rules to a full set of complex rules authored by domain experts and rules produced by other the state of the art induction algorithms.

2. BACKGROUND

2.1 Argument Diagrams

Argument diagrams are semi-formal graphical representations that reify key features of arguments such as *hypothesis* statements, *claims*, and *citations* as nodes and the *supporting*, *opposing*, and *clarification* relationships between them as arcs. Argument diagrams directly connect the syntax of the argument representation to the underlying semantics thus making it clear and computationally tractable. Argument diagrams can serve to make the often implicit structure of an argument *salient* to students while also *constraining* them to make relevant contributions [29]. Prior researchers have shown that argument diagrams can be used to scaffold students' understanding of existing arguments [12, 8]; can frame collaborative learning [26]; and can help to support scientific reasoning [29].

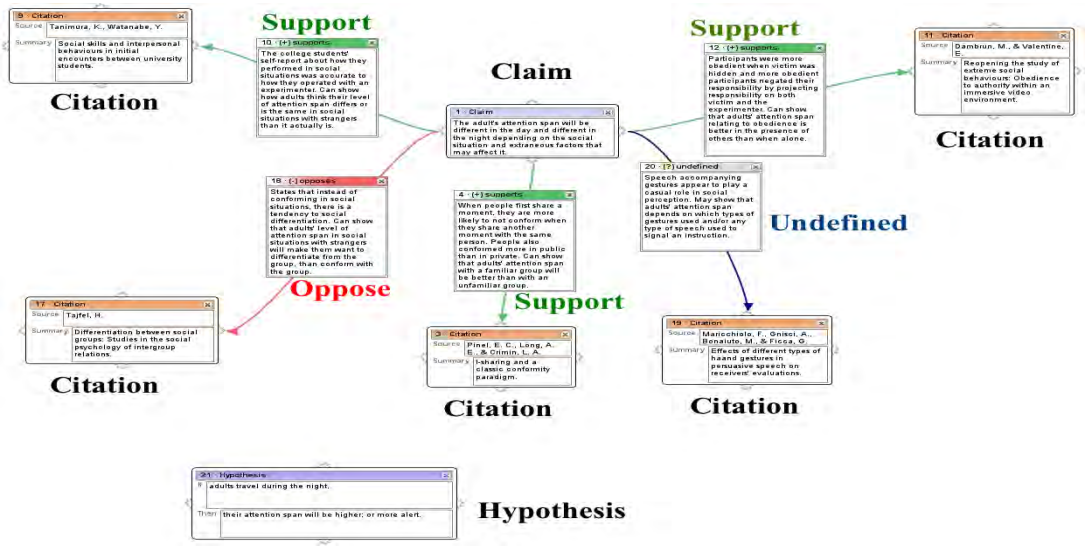


Figure 1: A student-produced Argument Diagram.

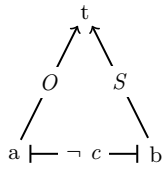
A sample student-produced diagram is shown in Figure 1. The diagram includes a central research *claim* node, which has a single text field indicating the content of the research claim. A set of citation nodes are connected to the claim node via supporting, opposing and undefined arcs colored green, red, and blue respectively. Each citation contains two fields: one for the citation information, and the other for a summary of the work; each arc has a single text field explaining what purpose the relationship serves. At the bottom of the diagram, there is a single isolated hypothesis node that contains two text fields, one for a conditional or *IF* field, and the other for a consequence *THEN* field.

2.2 Augmented Graph Grammars

Graph Grammars are a graph-based representation for rules about graphs that are analogous to string grammars. Graph grammar rules are composed of standard graph elements such as nodes and directed or undirected arcs. As with string grammars they are defined by a finite alphabet of basic or *ground* node and arc types as well as a set of production rules for *variable* elements. A single graph *rule* defines a space or *class* of matching graphs. Graph grammars can be used to generate graphs from an initial seed via recursive rule applications where each variable element expands to a larger subgraph. They can also be used to match graphs in a layered fashion by first mapping all ground elements to individual nodes or arcs and then recursively matching the sub-elements. Graph grammars have been used for analysis and graph transformation in domains such as visual programming [9] and mechanism analysis [27].

Augmented Graph Grammars are an extension of traditional graph grammars that are allow us to match *rich graphs* with complex node and arc types that contain sub-elements, text, and other variable structures [15]. Augmented Graph Grammars also support: negated elements which select for the nonexistence of subgraphs; generalized node and arc types

$(ParedWcomp)$



$$\left\{\begin{array}{c} t.Type = \text{“}claim\text{”} or \text{“}hypothesis\text{”} \\ a.Type = \text{“}citation\text{”} \\ b.Type = \text{“}citation\text{”} \\ c.Type = \text{“}comparison\text{”} \end{array}\right\}$$

Figure 2: A simple augmented graph grammar rule that detects uncompared counterarguments.

which match multiple items; complex element constraints which allow us to compare individual elements; complex graph expressions which allow for universal and existential quantification; and the incorporation of NLP rules or other external features. As such they are an ideal rule representation for the analysis of argument diagrams, user-system interaction logs, and other educational data.

A sample rule is shown in Figure 2. This rule is designed to identify cases of uncompared counterarguments, that is: there is an opposing arc O from the citation a to the node t and also a supporting arc S from the citation b to the node t, however, there exists *no* comparison arc between the two citations a and b. This is designated by the negated arc $\neg c$. Here node t is either a claim or hypothesis. The variable elements O and S are defined by recursive production rules which are not shown. Those rules define supporting paths as chains of supporting arcs and opposing paths as chains of supporting arcs with any odd numbered (including single) chain of opposing arcs.

This example rule was designed by a domain expert in argumentation. It is designed to identify cases where a student has presented conflicting background information but has made no attempt at resolution. This is a critical structural flaw that is commonly found in student-produced arguments. Students at all levels frequently absorb the lesson that they must show conflicting citations but routinely fail to explain those citations or to resolve the differences in a way that clarifies their own argument. As we have shown previously such expert-designed rules can be empirically-valid and predictive of student performance [16]. However manually designing rules can be both costly and inefficient.

Thus our goal is to automatically induce meaningful rules, rules that highlight structural flaws or argumentation errors; rules that generalize beyond basic types; and rules that include negated elements(detecting non-existing cases).

2.3 Graph Grammar Induction

Current grammar induction algorithms fall into one of two broad categories: frequent subgraph matching, or graph compression. Frequent subgraph algorithms include Yan and Han's gSpan algorithm [32], Inkokuchi's AGM [1], and the FSG algorithm [20]. These algorithms carry out controlled graph walks to identify common structures. They are quite effective, particularly in grounded domains such as cheminformatics where the graphs, in this case molecular models, have low degree and exact matches are required. However the algorithms do not support disjoint subgraphs, negation, or generalized elements. While we can, in theory, insert explicit negation arcs that would expand the size of the graphs exponentially and thus make any search process intractable. Similarly, while we could replace individual elements with generalized forms that would simply force the system to use a smaller range of types and would not allow for context-sensitive generalization of elements. These algorithms are also ill-suited for identifying errors as the search process is strictly unsupervised and finds frequently-occurring structures without reference to external weights.

Graph compression algorithms such as Subdue take a different approach to the problem. Subdue is a recursive beam-search algorithm that generates a hierarchical grammar by recursive collapse based upon the MDL principle [5]. Subdue operates by iteratively identifying the most frequently occurring arc in the graph and then reducing it to a new variable node. Unlike gSpan the resulting grammar is hierarchical and the beam search process can be used for supervised learning given a suitable set of positive and negative examples [11]. The candidate graphs are ranked according to a normalized error metric:

$$\frac{(PosGraphsNotCovered + NegGraphsCovered)}{TotalExamples}$$

While Subdue is more flexible than the frequentist approaches it too does not support generalized elements, negation, or disjoint subgraphs.

2.4 Related Work

We have previously shown that domain experts can hand author augmented graph grammars that are empirically-valid and which can be used as features in a regression model to automatically grade student-produced diagrams [16, 17]. In more recent experiments we have also shown that it was possible to apply EC to induce graph grammars that are positively correlated with argument grades and that we can apply χ^2-filtering to select unique rules from the large space of candidates [19]. We were also able to show that the induced rules outperformed rules generated by both Subdue and gSpan and outperformed similar expert rules that fit into the limited rule space. The rules produced in that study, however, were limited in scope. While they supported disjoint graphs, they did not identify errors, and did not support generalized elements or negation. In this work we will build upon these results to include generalization and negation, and we will compare the resulting rules to a full set of 77 hand-coded expert rules.

3. METHODS

We conducted two experiments on the induction of Augmented Graph Grammars using EC. First we applied EC to induce graph rules composed of static node and arc types that were both positively and negatively correlated with the overall argument quality. That is, we sought to identify ground rules that either highlighted good features of arguments **(positive)** or matched structural flaws**(negative)**. We then compared them to expert-produced rules and to rules induced by the Subdue and gSpan algorithms. In our second experiment we applied EC to induce rules that also incorporated generalized nodes as well as negated arcs **(detecting non-existing cases)**. We describe them below.

Evolutionary Computation is a general beam-search algorithm based upon Natural Selection. The EC algorithm begins with a population of candidate solutions in a shared solution representation. This population may be randomly generated or supplied by the user. The individual solutions are then ranked by means of a fitness function which may be an absolute performance metric or a form of tournament selection. The next generation of the population is then formed by a combination of fitness proportional selection, crossover or recombination of candidate solutions, random mutation of solutions, and elitist cloning. EC algorithms proceed iteratively until a given fitness threshold is reached or a fixed number of generations has passed. EC has been used in a number of applications such as tuning Neural Networks [21], and evolving computer code [2].

EC has a number of advantages over other special-purpose induction algorithms. Firstly, it is very flexible, the behavior of the system is determined by the user-specified solution representation and the genetic operators. This makes it easy to tune the behavior of the system to include new types of elements or to test out alternative inductive biases. Secondly, EC is very robust, the basic algorithm can be applied in a wide range of domains and it can be used in areas where the contours of the search space is unknown. There are a number of widely-available EC systems. For the purposes of this research we used *pyEC* an open-source EC engine [18] coupled with *AGG* an engine for graph matching using Augmented Graph Grammars [15].

The rules induced in Experiment I consisted entirely of ground nodes and arcs while the rules induced in Experiment II included generalized node types and negated comparisons as shown in Figure 2. For both experiments we assessed the

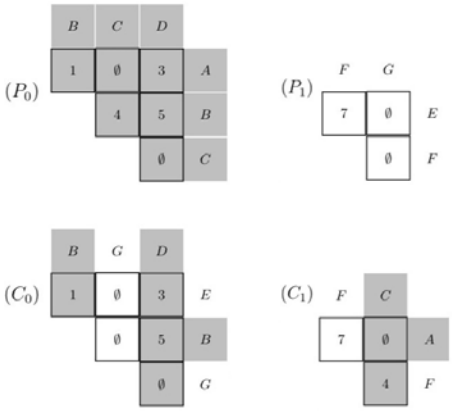


Figure 3: Canonical matrices for crossover.

fitness of the rules using the same nonparametric frequency correlation that we discussed in Subsection 2.4 with the target values being maximized or minimized depending upon the experimental goals.

Mutation in the EC algorithm is a general-purpose operation that is designed to promote exploration by introducing heterogeneity into the population. For this set of experiments we applied basic point mutation that added, deleted, or modified individual graph elements (see [33, 19]). Here mutation occurred with a small constant frequency when individuals were added to each population.

For these experiments we employed stable matrix crossover based upon the work of Stone, Pillmore, & Cyre [28] illustrated in in Figure 3. In this form of crossover we select a pair of parent graphs using fitness-proportional selection and represent them as adjacency matrices (P_0). The nodes are represented by letters on the rows and columns, while the arcs are represented by the numbered cells within the table. Empty cells indicate the absence of an arc. The order of elements in the matrices is canonical and is determined by the order in which the nodes were added to the rule.

On crossover we align the nodes and arcs in the parent matrices and then randomly shuffle the nodes and arcs between them based upon a series of coin tosses to produce the two children (C_0). Any constraints that are attached to an individual element are copied with it. Matrix crossover always produces two children that match the size of their parents with all excess elements being copied directly to the larger of the two offspring. Table 1 shows this crossover process at the graph level. By design crossover is an adaptive process that is designed to promote homogeneity and to preserve good building blocks or partial solutions called *introns* [2].

4. DATA

Our experimental analysis was based upon two previously-collected datasets. The first is a set of student-produced argument diagrams for empirical research reports. The second is a repository of hand-authored rules defined by domain experts. Both datasets were collected as part of our prior work on the diagnosticity of argument diagrams [16, 17].

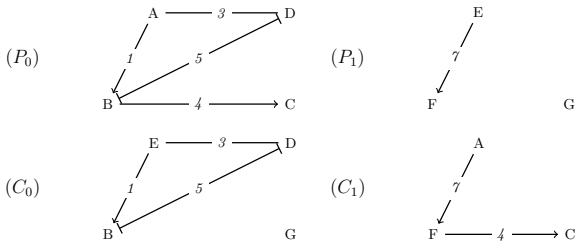


Table 1: Graphical representation for crossover.

4.1 Argument Data

Our repository of argument diagrams was collected at the University of Pittsburgh in a course on Psychological Research Methods. Students in the course learn about designing, conducting, and reporting on empirical research. The course has a significant writing component. Students complete two research projects over the course of the semester both of which result in a written report modeled on a conference publication. They are allowed to work on the projects individually or as a team of two. For the purposes of our study, the students were required to plan their written arguments graphically before writing them. The diagrams were authored using LASAD, an online tool for argument diagramming and collaboration [14]. The diagramming ontology contained four types of nodes: *citation*, *claim*, *current study* and *hypothesis*; and four types of arcs: *supporting*, *opposing*, *comparison*, and *undefined*. Currstudy nodes are used to represent factual information about the study such as the target population. Undefined arcs represent cases where nodes provide clarification or concept definitions.

After removing dropouts and one diagram containing a single node, we collected a set of 104 paired diagrams and essays from the course. These diagrams and essays were independently graded by an experienced TA according to a parallel rubric with 14 questions that were focused on the argument's quality, coherence, use of citations, and other criteria. In this work we will focus on the *gestalt* grades for overall graph and essay quality. The gestalt grades were assigned on an 11 point scale from -5 (worst quality) to +5 (complete, coherent, and persuasive) at $\frac{1}{2}$ point intervals. This same dataset was used in our prior work [19].

4.2 Expert rules

In parallel with data collection, we also collaborated with a group of domain experts to define a set of 77 *a-priori* argument rules. These rules were designed to identify individual features of argument diagrams or sub-graphs that were consistent with high quality argumentation or which represented structural flaws. Thirty-four of these rules focused on basic features such as the size or order of the diagram, the average number of parents and children, or the presence of empty elements. The remainder were complex rules that described the relationship between elements or matched larger graph structures such as the uncompared counterarguments shown in Figure 2. These rules included features that dealt with the text inside the elements, appropriate grounding of hypotheses or claims in citations, connectedness of the diagram, and the appropriate use of individual elements.

In prior work we evaluated whether or not these rules were *empirically-valid*. That is whether or not they correlated with the independently-assigned diagram grades and whether or not they could be used to predict the paired essay grades [16, 17]. In that work we assessed the validity of each individual rule by testing the correlation between the observed rule frequency on each diagram and the final graph or essay grade. The strength of this correlation was assessed using Spearman's ρ a nonparametric correlation measure [31]. We found that most, but not all of the rules were strongly correlated with the grades. We also found that some of the correlations ran counter to the experts' a-priori expectations.

5. EXPERIMENTS

In this work we induced sets of baseline rules using the Subdue and gSpan algorithms. We also conducted two sets of evolutionary experiments designated *EC-Base* and *EC-General*. The rules from each of these experiments were compared to assess their overall performance.

Subdue: For these experiments we used Subdue V5 [4] in supervised learning mode to induce rules that were positively and negatively correlated with the overall graph and essay grades. In order to induce positively correlated rules we partitioned the graphs into positive and negative examples based upon their graph or paired essay score. All graphs with a grade of 0 or more were treated as positive examples, and all graphs with a negative grade were treated as negative examples. We then ran the system to extract the 12 best rules. In order to induce negatively-correlated rules we reversed the assignment with rules that were graded less than or equal to 0 being treated as positive examples and all others being treated as negative. We experimented with more restrictive thresholds > 0 and < 0 and found the performance did not improve.

gSpan: In this experiment we used gSpan v6 [34]. The software runs in strictly unsupervised mode where it returns all subgraphs whose frequency exceeds a user-specified threshold. In this case we ran the software over our dataset and collected all rules that exceeded a 1% threshold and then ranked the candidate rules based upon their ρ value to identify the most positive and negative examples.

EC-Base: In this experiment, we conducted a series of six evolutionary runs that were tuned to induce negatively-correlated rules. Three of those runs used the graph grade as a target and three used the essay grade. In each case we used a fixed population size of 100 individuals and ran the algorithm for 1,000 generations. In each generation, we cloned the top 10 individuals directly into the next generation under elitism. We selected 10 individuals for point mutation and the remaining 80 individuals for crossover, then we copied the results over to the next generation. Fitness values were assigned using a fixed measure of $-\rho$ for each individual rule. The initial populations were composed of randomly-generated individuals containing 3 - 10 elements each. The nodes and arcs were all ground elements and were selected from a predefined ontology of basic types that matched the types used in the argument diagrams.

Unlike standard EC we did not rely solely on the final population of rules for our results. EC populations grow increasingly homogeneous over time making the final population virtual clones. In this case our goal was to induce a range of potential rules. We therefore collected candidate rules from each generation of the run by selecting every rule with a $\rho \leq -0.1$. The full set was used in our analysis.

EC-General: Here we conducted a series of twelve evolutionary runs. Six of the experiments were tailored to induce positively correlated rules while the rest were tailored to induce negatively-correlated ones. As with EC-Base the population size was 100, the algorithm ran for 1,000 generations, and we used $\pm\rho$ as the basic fitness metric and the mutation and crossover rate were the same as before. Unlike the EC-Base study these rules also included negated comparison arcs as well as two generalized node types: nodes that are citations or claims (*CitOrClaim*) and nodes that are hypotheses or claims (*HypOrClaim*). These elements were chosen for addition because they were used by the domain experts when crafting their rules. As before we collected candidate rules from the positive and negative runs with thresholds of ($\rho \geq 0.18$) and ($\rho \leq -0.1$) respectively. These thresholds were chosen based upon a series of exploratory runs in which we found that the ρ values became statistically significant after exceeding ± 0.18.

6. RESULTS & ANALYSIS

Table 2 shows the number of positively and negatively correlated rules for the Graph grades (columns 3 and 4) and the Essay grades (columns 5 and 6) that were collected during our experiments. *Total* designates the total number of rules produced by each method or in the expert set, while *Threshold* indicates the number for which $\rho \geq 0.18$ or $\rho \leq -0.18$ in the positive and negative cases respectively.

As Table 2 shows the EC approaches generated the largest number of candidate rules in both the positive and negative cases. Of the expert rules, most of them were positively correlated with performance but less than half of them exceeded the cutoff thresholds. Indeed only two of the expert rules did so for the essay grades. Both Subdue and gSpan identified positively and negatively-correlated rules but only a few of the positive rules exceeded the threshold. None of the negative rules did so.

Next, we will describe the rules induced during our EC-Base

Table 2: Number of Positive and Negative Rules

Methods		Graph		Essay	
		Pos	Neg	Pos	Neg
Subdue	Total	12	2	8	10
	Threshold	11	0	3	0
gSpan	Total	34	5	27	12
	Threshold	12	0	6	0
Expert	Total	56	21	46	32
	Threshold	25	6	0	2
EC-B	Total	82	256	172	160
	Threshold	82	51	172	22
EC-G	Total	394	392	652	518
	Threshold	394	193	652	30

$\star$ Threshold: number of rules with $\rho \geq 0.18$ or $\rho \leq -0.18$

Table 3: Spearman correlation values for the best 3 rules in each experiment.

	Positive-correlated						Negative-correlated					
	Graph			Essay			Graph			Essay		
	1st	2nd	3rd	1st	2nd	3rd	1st	2nd	3rd	1st	2nd	3rd
Subdue	.276	.270	.253	.281	.215	.181	-.050	-.022	NA	-.173	-.167	-.164
gSpan	.352	.314	.272	.300	.281	.261	-.137	-.063	-.05	-.123	-.102	-.075
Expert	**.427***	.338	.329	.180	.138	.137	-.238	-.236	-.202	**-.256**	-.218	-.148
EC-B	.371	**.369**	**.362**	**.334**	**.334**	**.319**	**-.272**	**-.272**	**-.271***	-.233	**-.233**	**-.233**
EC-G	.396	**.391***	**.385***	**.357***	**.357***	**.356***	**-.273***	**-.272***	-.270	**-.269***	**-.269***	**-.269***

⋆ The best of results for Experiment I is in bold;
⋆ '∗' is for best of results across both Experiment I and II.

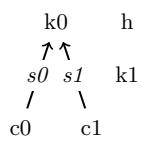
experiment and we will discuss how they compare to the expert rules and the rules induced by Subdue and gSpan. We will then discuss the EC-General rules and compare them to our earlier results.

6.1 Experiment I: EC-Base

Rows 1-4 in Table 3 list ρ values for the three best rules from the four methods. The bold values indicate the best performing rule among the sets. As the table illustrates EC-B outperformed both Subdue and gSpan across the board. And it outperformed the expert rules in most cases. The lone exception being the best positive case for the graph grades and the best negative case for the essay grades.

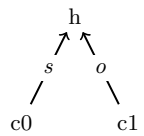
The best positively-correlated expert rule for the graph grades matched arcs with empty text fields. The best negatively-correlated expert rule with the essay grade matched graphs with no hypothesis nodes. Both of these rules relied on complex grammar features, textual rules and expressions, that were outside the scope of our current experiments.

(B-G-P)



$$\left\{\begin{array}{c} k*.Type = \text{“}claim\text{”} \\ h.Type = \text{“}hypothesis\text{”} \\ c*.Type = \text{“}citation\text{”} \\ s*.Type = \text{“}supporting\text{”} \end{array}\right\}$$

(B-E-P)

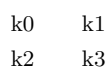


$$\left\{\begin{array}{c} h.Type = \text{“}hypothesis\text{”} \\ c*.Type = \text{“}citation\text{”} \\ s.Type = \text{“}supporting\text{”} \\ o.Type = \text{“}opposeing\text{”} \end{array}\right\}$$

Figure 4: EC-Base: Strongest Positively-correlated Rules Induced by EC.

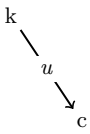
Figures 4 and 5 illustrate the best positive and negative rules induced by the EC-Base runs. In Figure 4 graph rule B-G-P represents a rule that has 5-nodes, two of which are citations ($c0$ & $c1$) that support a shared claim node ($k0$). The remaining nodes are a single claim ($k1$) and a hypothesis (h) which may or may not be connected to the rest of the structure. This reflects a graph where the authors identified at least two related citations that can be synthesized to support a single claim and where they included both a hypothesis and another claim. This is one of the structures

(B-G-N)



$$\left\{\begin{array}{c} k*.Type = \text{“}claim\text{”} \\ cs*.Type = \text{“}currstudy\text{”} \end{array}\right\}$$

(B-E-N)



$$\left\{\begin{array}{c} k.Type = \text{“}claim\text{”} \\ c.Type = \text{“}citation\text{”} \\ u.Type = \text{“}unspecified\text{”} \end{array}\right\}$$

Figure 5: EC-Base: Stronges Negatively-correlated Rules Induced by EC.

that students have been encouraged to make in their arguments as it shows an ability to synthesize citations to form a complex claim.

Interestingly, the best positive essay rule (B-E-P) is very closely related to the expert rule shown in Figure 2. Here it selects for the presence of a hypothesis node (h) that is directly connected to two citations ($c0$ & $c1$). Here $c0$ directly supports h while $c1$ directly opposes it. Given that the algorithm could not induce variable arcs it is not surprising that it does not include paths. The absence of a comparison arc, however, is interesting. As we noted above the students were instructed to include one. The fact that this rule performs so well despite lacking one suggests that the students did not regularly do so.

Figure 5 shows the best negative rules. As stated above, we expect that these rules will flag errors or persistent structural flaws. B-G-N consists of 4 claim nodes ($k0 - k3$) and two currstudy nodes ($cs0$ & $cs1$) all of which may or may not be connected to one-another. While this rule has a high correlation with the grade, its semantic meaning is unclear. It is possible that it is detecting is overly large graphs that lack sufficient focus. In future work we will evaluate the matching graphs with domain experts to assess this.

B-E-N is easier to interpret. In this case the rule contains a single claim node (k) which is connected to a citation node (c) via an undefined arc (u). This is a clear violation of the semantic guidance that students were given. The students in the experiment were instructed to use unspecified arcs for definitions or clarifications only. Some students instead

used them when they were unsure about the strength of their evidence or did not understand the citation. The students were also instructed to use citations to add information to their claims, not the other way around. For a student to use an unspecified arc in this way suggests that they were unsure about the structure or content of the argument.

6.2 Experiment II: EC-General

The last row of Table 3 shows the performance of the EC-General rules. These rules were compared against all of the rules in Experiment 1. The best performing rules across both experiments are in bold and marked *. As Table 3 shows EC-General produced better performing rules than EC-Base. All but one of the ρ values on the final row exceeds the corresponding value on the fourth, and the one that does not do so falls behind by only 0.001. EC-General outperformed the best negative expert rule for the essay grades (-0.269 vs. -0.256), despite the fact that the expert rule relied on complex expressions. The best expert rule for the graph grade still outperforms EC-General. Thus, our results for EC are better than all other methods save for one expert rule that relies on novel textual features.

Figure 6 shows the best positively-correlated rules for the graph and essay grades. G-G-P matches cases where a supporting arc has been drawn from a citation or claim to a claim or hypothesis. In short, it matches correct uses of supporting arcs. This is a good feature that indicates well-supported arguments. G-E-P, by contrast, is complex and selects for a graph with three claim nodes ($k0 - k2$) and two uncompared citations ($c0$ & $c1$), where $c1$ directly supports a hypothesis or claim (hk) which in turn has an unspecified arc to a citation or claim node (ck). The semantic meaning of this rule is unclear and will require deeper analysis.

Figure 7 shows the strongest negatively-correlated rules. As with G-E-P, G-G-N, is somewhat hard to interpret. It selects for a number of disjoint nodes, and for the presence of a currstudy node ($cs0$) as well as a claim ($k3$) which are not connected via a comparison arc. Further analysis is required to determine why this rule holds. G-E-N, by contrast represents a clear variation on B-E-N. Here we select for a hypothesis or claim node (hk) that has an undefined arc to a citation along with a separate hypothesis node that may or may not be connected. This rule is interesting because in part it will select a superset of the graphs matched by B-E-N but the presence of the extra hypothesis node will restrict that somewhat. This suggests that this rule may be relatively specific to our dataset. We plan to examine the matching graphs to assess its generality.

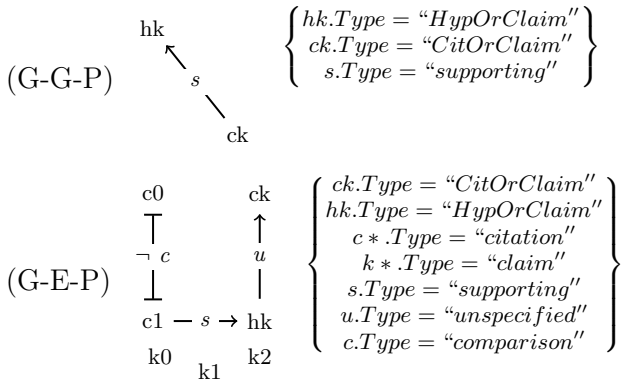


Figure 6: EC-General: Strongest Positively-correlated Rules Induced by EC.

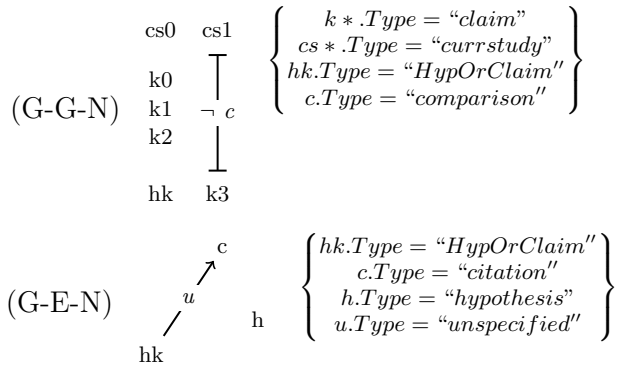


Figure 7: EC-General: Strongest Negatively-correlated Rules Induced by EC.

7. CONCLUSIONS

In this paper, we reported our work on the automatic induction of Augmented Graph Grammars for student-produced argument diagrams through EC. In prior work we demonstrated that hand-authored expert rules can be empirically-valid and that those valid rules can be used for automatic grading. We have now shown that it is possible to automatically induce complex rules for argument diagrams that match both positive and negative examples and which can therefore be used as features for automatic grading. We have also shown that the induced rules outperform all but one of the expert rules and the rules induced by other general-purpose grammar induction algorithms. The strongest expert rule was outside the scope of this experiment.

In future work we plan to work with domain experts to evaluate these rules. Our goal will be to determine whether the rules are semantically valid, and whether or not they can serve as the basis for automatic hints. We will also assess whether or not the rules can be used for data-driven grading by using them as features in a regression model. And finally we will expand the scope of our EC induction to include the automatic induction of hierarchical rules with expressions and complex element constraints.

8. REFERENCES

[1] I. Akihiro, W. Takashi, and M. Hiroshi. An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery*, pages 13–23. Springer, 2000.

[2] W. Banzhaf. *Genetic programming: an introduction on the automatic evolution of computer programs and its applications*. ACM, 1998.

[3] N. Belacel, G. Durand, and F. LaPlante. A binary integer programming model for global optimization of learning path discovery. In Santos and Santos [25].

[4] D. Cook, L. Holder, J. Coble, S. Djoko, B. Eberle, G. Gelal, J. Gonzalez, I. Jonyer, and N. Ketkar. Subdue version5, 2012.

[5] D. J. Cook, L. B. Holder, and N. Ketkar. Unsupervised and supervised pattern learning in graph data. In *Mining Graph Data*, chapter 7, pages 159–180. John Wiley & Sons, Inc, Hoboken, New Jersey, 2007.

[6] J. Dai., R. B. Raine., R. Roscoe., Z. Cai., and D. S. McNamara. The writing-pal tutoring system: Development and design. *Journal of Engineering and Computer Innovations*, 2:1–11, 2010.

[7] M. Eagle, A. Hicks, B. W. P. III, and T. Barnes. Exploring networks of problem-solving interactions. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge, LAK*, pages 21–30. ACM, 2015.

[8] M. W. Easterday, J. S. Kanarek, and M. Harrell. Design requirements of argument mapping software for teaching deliberation. In *Online Deliberation: Design,Research, and Practice. Stanford*, pages 317–323. CA: CSLI Publications/University of Chicago Press, 2009.

[9] K. A. et al. Graph grammar induction on structural data for visual programming. In *Applications of Graph Transformations with Industrial Relevance*, volume 7233, pages 121–136, 2012.

[10] R. S. B. et al. A MOOC on Educational Data Mining. In S. ElAtia, O. R. Zaiane, and D. Ipperciel, editors, *Handbook of Data Mining and Learning Analytics*. Hoboken, NJ: Wiley, 2016. (in press).

[11] J. A. Gonzalez, L. B. Holder, and D. J. Cook, editors. *Graph-Based Concept Learning*, Florida,USA, 2001. AAAI.

[12] M. Harrell and D. Wetzel. Improving first-year writing using argument diagramming. In *Proceedings of the 35th Annual Conference of the Cognitive Science Society*, pages 2488–2493, 2013.

[13] K. R. Koedinger, J. C. Stamper, B. Leber, and A. Skogsholm. LearnLab's DataShop: A data repository and analytics tool set for cognitive science. *Topics in Cognitive Science*, 2013.

[14] F. Loll and N. Pinkwart. Lasad: Flexible representations for computer-based collaborative argumentation. *International Journal of Human-Computer Studies*, 71:91–109, Januart 2013.

[15] C. F. Lynch. Agg: Augmented graph grammars for complex heterogeneous data. In Santos and Santos [25].

[16] C. F. Lynch and K. D. Ashley. Empirically valid rules for ill-defined domains. In J. Stamper and Z. Pardos, editors, *Proceedings of The 7th International Conference on EDM 2014*. IEDMS, 2014.

[17] C. F. Lynch, K. D. Ashley, and M. Chi. Can diagrams predict essay grades? In S. Trausan-Matu, K. E. Boyer, M. E. Crosby, and K. Panourgia, editors, *Intelligent Tutoring Systems*, Lecture Notes in Computer Science, pages 260–265. Springer, 2014.

[18] C. F. Lynch, K. D. Ashley, N. Pinkwart, and V. Aleven. Argument graph classification with genetic programming and c4.5. pages 137–146.

[19] C. F. Lynch, L. Xue, and M. Chi. Evolving augmented graph grammars for argument analysis. Genetic and Evolutionary Computation Conference, 2016.

[20] K. Michihiro and K. George. Frequent subgraph discovery. In *Proceedings IEEE International Conference on Data Mining. (ICDM 2001)*, pages 313–320. IEEE, 2001.

[21] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press: Cambridge Massachusetts, 1999.

[22] N. Pinkwart, K. D. Ashley, C. F. Lynch, and V. Aleven. Evaluating an intelligent tutoring system for making legal arguments with hypotheticals. *International Journal of Artificial Intelligence in Education (IJAIED)*, 19(4):401 – 424, 2009.

[23] K. Porayska-Pomsta and K. Verbert, editors. *Workshops Proc. 8th International Conference on Educational Data Mining, EDM 2015*, volume 1446 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[24] A. Poulovassilis, S. G. Santos, and M. Mavrikis. Graph-based modelling of students' interaction data from exploratory learning environments. In Porayska-Pomsta and Verbert [23].

[25] S. G. Santos and O. C. Santos, editors. *Proceedings of the 7th International Conference on Educational Data Mining (EDM 2014)*, volume 1183 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014.

[26] O. Scheuer, B. McLaren, F. Loll, and N. Pinkwart. Automated analysis and feedback techniques to support argumentation: A survey. In N. Pinkwart and B. M. McLaren, editors, *Educational Technologies for Teaching Argumentation Skills*. Bentham Science Publishers, 2012. (in press).

[27] L. C. Schmidt, H. Shetty, , and S. C. Chas. A graph grammar approach for structure synthesis of mechanisms. *Journal of Mechanical Design*, 122:371–376, 1999.

[28] S. Stone, B. Pillmore, and W. Cyre. Crossover and mutation in genetic algorithms using graph-encoded chromosomes. *Unpublished*, March 2011.

[29] D. D. Suthers. Empirical studies of the value of conceptually explicit notations in collaborative learning. In A. Okada, S. Buckingham Shum, and T. Sherborne, editors, *Knowledge Cartography*, pages 1–23. Springer Verlag, 2008.

[30] G. Wang, Y. Han, Z. Zhang, and S. Zhang. A dataflow-pattern-based recommendation framework for data service mashup. In *Services Computing, IEEE Transactions*, volume 8, pages 889 –902. IEEE, 2015.

[31] Wikipedia. Spearman's rank correlation coefficient — wikipedia, the free encyclopedia, 2013.

[32] Y. Xifeng and H. Jiawei. gspan: Graph-based substructure pattern mining. In *Proceedings of the IEEE International Conference on Data Mining (ICDM 2002)*, pages 721–724. IEEE, 2002.

[33] L. Xue, C. F. Lynch, and M. Chi. Graph grammar induction via evolutionary computation. In Porayska-Pomsta and Verbert [23].

[34] X. Yan. gspan: Frequent graph mining package, 2009.

Short Papers

Investigating Swarm Intelligence for Performance Prediction*

Mohammad Majid al-Rifaie†
Goldsmiths, University of London
Department of Computing
London SE14 6NW, UK
m.majid@gold.ac.uk

Matthew Yee-King
Goldsmiths, Uni of London
Department of Computing
London SE14 6NW, UK
m.yee-king@gold.ac.uk

Mark d'Inverno
Goldsmiths, Uni of London
Department of Computing
London SE14 6NW, UK
dinverno@gold.ac.uk

ABSTRACT

This paper proposes a new technique for analysing the behaviour of students on an online course. This work considers a range of social learning behaviours supported in our recently designed and implemented collaborative learning system which supports students giving and receiving feedback on each other's developing work and practice. The course was delivered to several thousand students on Coursera during which students were directed onto our social learning environment to take part in group work and assessment activities. This work introduces a swarm intelligence technique, Stochastic Diffusion Search (SDS), and shows how it can be adapted and applied to our data in order to perform classification tasks. The novelty of the approach is not only in using this technique, but also applying it to data linked to *social behaviour* (i.e. how students interact with each other) which differentiates the work apart from many clickstream analysis studies. This paper investigates what combined activity is the best predictor of success or failure in the course. The aim is to argues that the results obtained using the proposed approach indicate the promising potential of predicting students performance through applying swarm intelligence technique to social behaviours. This work has a number of potential benefits including designing better social learning systems, designing more effective social learning and assessment exercises, and encouraging disengaged students. In addition, this work is an important step in addressing our long term goal of evidencing how critical student learning takes place as they give and receive feedback to and from each other on work in progress.

Keywords

Social learning, swarm intelligence, education system modelling, MOOC

1. INTRODUCTION

Increasingly researchers are focusing on the significance of social learning and investigating its impact within the various online learning environments. Acknowledging the importance of collaboration and 'teamwork', as an embedded element in the Massive Open Online Courses (MOOCs), this method of learning is desirable for many employers who rely on highly collaborative and online-based works. Our programme of work is concerned with designing a novel learning technology, online courses and assessments, which provide us with a range of data we can use to understand how learning takes place through online social interaction. Our pedagogy is influenced by our home institution's "art-school" pedagogy across practice-based subjects (such as art, music and design) where students learn by sharing "work in progress" within tutor groups and giving and receiving feedback to each other. The aim of this work is to use learning analytics to build strong arguments for the adoption of social learning pedagogies supported by innovative technology. Therefore this paper focuses on extracting information from *social learning activity logs*, not the full range of more traditional courseware access and activity logs. The objective is to gain a better understanding if these activities have any measurable relation to learning, and if so which are the most important activities and in which combinations. The analysis presented here is a first step in that direction, where the attempt is to predict if students will pass or fail a course, using only low level user interface telemetry data gathered from our social learning platform. Given the undeniable significance of data classification in different and diverse scientific domains (e.g. computer science, psychology, medicine), various techniques have been proposed over the years. Nature-inspired metaheuristic algorithms are among one of the categories which aimed at providing solutions to this problem.

In this paper a novel method in addressing data classification in the context of educational data is used where a swarm intelligence algorithm is adapted for this purpose. A recent review [2] details the extensive applications of this algorithm in the last two decades in various fields (e.g. discrete and global optimisation, pattern recognition, resource allocation, medical imagining, etc).

*(Does NOT produce the permission block, copyright information nor page numbering). For use with ACM_PROC_ARTICLE-SP.CLS. Supported by ACM.

†Corresponding author

The research questions which drive this paper are as follows:

1. How can the proposed swarm intelligence technique (SDS) be applied to educational data?
2. What kinds of social learning activities, and what combinations of social learning activities are the best predictors?
3. Does social interaction data contain strong predictive potential of student success?
4. How does an SDS analysis of social learning data help us in designing and delivering learning activities, in improving social/group learning activities, and in building better social learning systems?

In this paper, first Stochastic Diffusion Search (SDS) algorithm is explained, detailing its behaviour and highlighting one of its main features (i.e. partial function evaluation). Then, an introduction is given to the classification problem in general followed by a brief section on the nature of the educational dataset used in this paper and the features available from the dataset. After elaborating on the data in the datasets in the context of the work, the swarm intelligence algorithm used is adapted for the purpose of the experiments conducted in this paper and the results are reported. A discussion on the behaviour of the proposed algorithm is presented showing its potential in using all the available features as well as identifying the most significant features. Finally, the paper is concluded with the summary of the research reported in the paper along with directions for future research.

2. RELATED WORK

With the increasing use of online learning platforms, a large number of researchers have been working on predicting grades from students performance over the course of the studies. This topic of research is of importance because, for example, only in the United States several hundred thousand students drop out of high school every year and perhaps interventions can provide the means to reduce the number of those falling behind in their studies [1, 7]. With the growing interest in MOOCs as alternative or adjunct learning platforms, behaviour prediction has attracted the attention of many educational data analyst, such as Brady et al. [15] who used higher granularity temporal information for their analytics work; in another work, Macfadyen et al. [8] explain the concept of "an early warning system" for educator, aiming to provide the means for the educators to intervene with an appropriate set of actions to improve the performance of the weaker students; a similar work was presented by Rogers et al. [11] which aims to identify students at rist of failure. The predictive power of demographics versus activity patterns in MOOCs are discussed by Brooks et al. [3] focusing on whether it is possible to find a link between performance and demographics. Other researchers, such as Coleman et al. [4] or Elbadrawy et al. [6], have also been exploring whether it is feasible to identify behavioural patterns for prediction. In addition to attempting to improve students performance, Yang et al. [14] have been focusing on the concept of dropouts which is a critical challenge for online courses. Considering the above recent work, it is evident that extracting useful knowledge from education data should ultimately be incorporated in the design of the online systems. In a recent work by researchers from Harvard University and MIT, Whitehill et al. [13] emphasised on the importance of intervention and especially automatic intervention in MOOCs in order to take measures to reduce the number of students quitting; they claim that their proposed system might encourage students to return into the course. In another work, by Rollinson and Brunskill, [12] emphasis has been put on the importance of coupling predictive models with an alternative student model and policy (which constitute the core of the Intelligent Tutoring Systems), focusing again on the importance of using predictive models along with other tools. Having mentioned the above research, it is important to state that arguably one of the important features in MOOCs is enabling learners to discuss their work with their peers and receive feedback. In a recent research, Olsen et al. [9] direct the prediction power towards collaborative learning environment; in their work, they argue that by adding collaborative learning features they were able to enhance their understanding on the impact of collaborative learning. Tightly related to the mentioned work, the importance of social centrality in the context of MOOCs is discussed by Dowell et al. [5] where they adopt an approach, which uses language and discourse as a tool to explore the association with the existing and established measures related to learning (i.e. traditional academic performance and social centrality). While this work does not endorse or reject the impact of social learning, it clearly shows an increasing interest in exploring the impact of collaborative learning.

3. STOCHASTIC DIFFUSION SEARCH

Stochastic Diffusion Search (SDS) [2] which was first proposed in 1989 is a probabilistic approach for solving best-fit pattern recognition and matching problems. SDS, as a multi-agent population-based global search and optimisation algorithm, is a distributed mode of computation utilising interaction between simple agents. Its computational roots stem from Geoff Hinton's interest in 3D object classification and mapping and its applications span from continuous optimisation to medical imagining. The SDS algorithm commences a search or optimisation by initialising its population and then iterating through two phases: the test and diffusion phases. In the test phase, SDS checks whether the agent hypothesis is successful or not by performing a hypothesis evaluation which returns a boolean value. Once the activity (i.e their status as being 'true' or 'false') of all the agents are determined, successful hypotheses diffuse across the population and in this way information on potentially good solutions spreads throughout the entire population of agents. In other words, each agent recruits another agent for interaction and potential communication of hypothesis. The spreading of information occurs during the diffusion phase.

In standard SDS (which is used in this paper), *passive recruitment mode* is employed. In this mode, if the agent is inactive, a second agent is randomly selected for diffusion; if the second agent is active, its hypothesis is communicated (*diffused*) to the inactive one. Otherwise there is no flow of information between agents; instead a completely new hypothesis is generated for the first inactive agent at random. Therefore, recruitment is not the responsibility of the active agents. In this work, activity of each agent is determined when its fitness is compared against a random agent (which is different from the selecting one); if the selecting agent has a better fitness (smaller value in minimisation problems)

Table 1: The list of features logged, along with examples of the total figures for a single student. The last column represents the grade correlation of each individual figure.

	Description	Example	Corr
F1	Play video	199	0.41
F2	Delete a reply	16	0.12
F3	Open item in search result list	0	0.15
F4	Report problem with media	22	0.48
F5	Load media	7580	0.41
F6	Report problem with reply	24	0.26
F7	Delete an annotation	0	0.19
F8	Save after edit	0	0.15
F9	View my files	954	0.40
F10	View set of shared files	8865	0.41
F11	Save after edit	0	0.11
F12	Delete video	0	0.18
F13	Periodically log and comment when video is playing	1928	0.30
F14	Play region and view thread	1313	0.53
F15	Save user profile	32	0.23
	Course final grade	100	1.00

than the randomly selected agent, it will be flagged as active, otherwise inactive. A higher rate of inactivity boosts exploration, whereas a lower rate biases the performance towards exploitation.

4. CASE STUDY AND DATASET

The analysis presented in this paper is based on a dataset gathered during a seven week creative programming course on Coursera which ran in Summer 2014. The course presented students with a series of worked example programs written using Processing [10] that were either musical, graphical or game based. It was assessed using weekly quizzes and three, biweekly peer assessments. The peer assessments required the students to select one of the tutor-supplied worked examples and extend it in some way of their choosing. They then had to create a five minute screencast video wherein they explained the changes they had made from the example code and demonstrated the running program. This video was uploaded to our social learning system and then a link to this was submitted to the main MOOC LMS. Our system allowed them to place comments along the timeline of the video and to view a range of suggested content from other students, such as highly commented and uncommented videos. Our system collects detailed logs of certain interface elements that the user clicked on or moused over, including a user id and a timestamp. The data set used in this paper consists of these clickstream logs plus final grades achieved on the course. There were a total of 993 students who created logs on our system and gained a final grade on the Coursera platform. The dataset spanned a period of about seven weeks. Each student's log data and final grade was converted into a feature vector containing totals for all of the observed log types taken over the entire time period of the study. Table 1 shows an example of such a vector. The research began by attempting to correlate individual elements of the vector to $final\_grade$ but individual correlations were statistically insignificant to predict grades so instead a multivariate classification approach is attempted, the results of which form the remainder of this paper. The main aim was to label students as pass (≥ 50) or fail (< 50).

5. APPLY THE SDS ALGORITHM

Here the process through which the SDS algorithm was adapted to perform the classification tasks is detailed and the steps taken during the *test* and *diffusion* phases are explained. In order to apply this swarm intelligence algorithm to the dataset the following are considered:

- **Search space** is the entire dataset
- **SDS hypothesis** refers to a student record
- **Student attributes:** Each student record has fifteen attributes or features (i.e. play, report_media, region_block, etc; see Table 1).
- **Micro-features:** The fifteen features of each student record are considered the micro-features of the hypothesis. Therefore each SDS hypothesis has fifteen micro-features referring to the attributes of the student.

Next, the phases used in SDS algorithm are highlighted and each phase is described briefly in the context of the dataset presented.

During the *initialisation phase*, one student is chosen randomly from the dataset and is set as a model. Then each agent is randomly associated with a student record from the search space. During the *test phase*, each agent (which is already allocated to a student) randomly picks one of the fifteen micro-features and compares its value against that of the model. If the difference between the two corresponding micro-features is within a specific threshold, τ_d (where τ is the threshold and d is the dimension) the agent becomes active, otherwise inactive. The process in the *diffusion phase* is the same as the one detailed in the algorithm description: each inactive agent picks an agent randomly from the population; if the randomly selected agent is active, the inactive agent adopts the hypothesis of the active agent (i.e. they refer to the same student as their hypothesis), otherwise the inactive agent picks a random student from the dataset.

Categories, Classes and Termination The agents iterate through the test and diffusion phases again until all agents are active. At this stage, the students referred to by all the active agents are assigned to a category. Additionally, the number of active agents on each student is logged. Once a category is determined, the process is repeated from the initialisation phase where agents are initialised throughout the search space and the first student which has not yet been assigned to any categories is set as the new model. Then the algorithm iterates through the test and diffusion phases until all students are allocated to a category. Finally, categories form the classes, and when there exist students that belong to more than one class, they will be allocated to the one which has attracted a larger number of active agents. The only tunable parameters for SDS is the swarm size, N which is empirically set to $N = 10,000$. Threshold, $\vec{\tau}$, which is the acceptable distance between the model and other samples for each dimension, d, is calculated using the following formula:

$$\vec{\tau}_d = \sum_{t=1}^{c} \left| \frac{\text{MAX}\left(\vec{I}_{id}^{t}\right) - \text{MIN}\left(\vec{I}_{id}^{t}\right)}{c} \right| \quad d = [1, 2, ..., 15] \quad (1)$$

where c is the number of student types or classes in the dataset (i.e. pass and fail); $\vec{I}_{id}^{t}$ represents the value of i^{th} student with type t and dimension d. There are 2 student types ($c = 2$) and the dimensionality of the problem is 15 (see Table 1). Therefore the difference between the minimum and maximum values in each band (e.g. pass and fail) is calculated, then the sum of the differences in each dimension is averaged and used to calculate the threshold. Using the formula above the threshold $\vec{\tau}$ is calculated using the

Table 2: Weekly breakdown of and fail/pass rate

	Wk1	Wk2	Wk3	Wk4	Wk5	Wk6	Wk7
Active students	245	974	629	683	488	528	265
Ratio	25%	98%	63%	69%	49%	53%	27%
% of fails	28%	39%	28%	16%	10%	5%	2%
% of passes	72%	61%	72%	84%	90%	95%	98%

training dataset. Using the threshold vector presented, if the randomly picked model falls on the first class (e.g. the fail class), it is likely that the active agents have a bigger presence in this class. It is worth noting that while in some iterations there is a high presence of active agents for some students, in some other iterations there is a high number of inactive agents on the same students. The reason why a student record could make an agent active in one iteration and inactive in another can be explained through SDS's random micro-features selection: each record consists of fifteen micro-features (the same as the number of attributes for each student), therefore if an agent picks one of the micro-features that are within the threshold, the agent becomes active, but if it randomly picks one of the other micro-features, the agent becomes inactive. Deducing from this, it is evident that having more micro-features within the range of the model results in more agents becoming (and staying) active, and as a result forming a stable category.

6. EXPERIMENTS AND RESULTS

In this section, the results of several experiments are reported along with a discussion on the relevance of the experiments to the research questions. The total number of students who used the online learning platform and obtained a final grade was 993. The number of active students each week and the fail/pass rate of students are detailed in Table 2, and the SDS algorithm is used as the classifier.

6.1 Experiment I: Weekly data analysis

The logged actions of all students who have participated in the previous and current weeks are cumulated and fed into the system for analysis.

One of the important elements in the cumulative data is the distribution of fail and pass in each of the training and test datasets. Fig. 1 shows this distribution in the test dataset. Note that the training datasets will have the same distribution as the test dataset. As illustrated in the figure, other than the first week, in the rest of the week, the cumulative data shows 39% and 61% of the data belonging to the fail and pass categories respectively. The classifier is trained and the prediction accuracy of the classifier is evaluated on the test datasets.

Table 3 and Fig. 2 show the weekly prediction-accuracy on the test datasets. As expected, and due to the presence of more data as students progress to the next weeks, there is a gradual increase in the prediction accuracy of the swarm intelligence classifier. Looking at the maximum value in Table 3, the prediction accuracy rises to 88% on week 7. The notable increase in the accuracy starts in week 4 (i.e. with median accuracy of 74% and the maximum accuracy of 80%, allowing the teachers to have a rough estimate about the students who are likely to pass or fail. The results reported in this paper are based on 30 runs for each experiment.

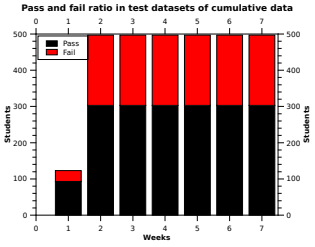


Figure 1: Pass / fail ratio in test datasets of cumulative data

Table 3: Weekly accuracy percentages

	Mean	Median	StDev	Min	Max
Week1	38.40	39	3.67	32	46
Week2	46.97	47	1.59	45	53
Week3	59.93	60	2.83	49	64
Week4	72.07	74	6.44	54	80
Week5	74.37	78	8.83	47	83
Week6	82.30	84.50	5.67	59	87
Week7	80.67	84	9.47	50	88

6.2 Experiment II: Analysis of feature vector

As highlighted before, one of the main purposes of analysing the presented data is identifying weaker students as early as possible and therefore finding ways of improving their performance. However, there are many features collected from the online learning platform and identifying the "more relevant" features from the entire feature vector (of size 15) is of importance. Therefore, each of the features, have been singled out and used both for training the swarm intelligence classifier as well as the evaluation phase. The summary of the solo performance of these features are reported in Fig. 3 and Table 4. For instance, feature 13 (F13 or 'playing') in all weeks (except week 1, 2 and 3) is the most influential feature and has returned the highest prediction accuracy. While the grade correlation of this feature is only 0.41, this finding highlights the role of watching videos in the learning process. Knowing what the feature represents, its value is evident and the algorithm proved capable of identifying this important feature. Identifying the most influential features would entail that the analysis could be focused on the n most important features, instead of stretching the computational power to consider all the input features for predication analysis. The results in this section demonstrate that there could exist some individual features which would provide stronger prediction power when used individually than along with the other features.

6.3 Experiment III: Feature combinations

As shown in Table 4, in order to identify the important features, the three most influential features in each week are labelled 1-3 in brackets. The impact of each feature is calculated by giving the weights of 6 to the most influential feature (shown as (1) in the table), and 3 and 1 to the second two influential features (shown as (2) and (3) in the table). The impact of each feature is then calculated using the aforementioned weights. The six most important features are listed below in the order of importance:

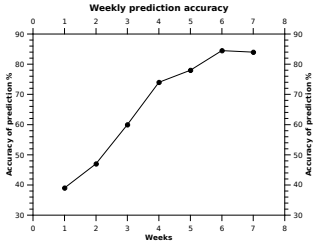


Figure 2: Prediction accuracy of the weekly cumulative data

Table 4: Analysing the impact of individual features (1-15). Prediction accuracies are shown in percentages. The three most influential features in each week are labelled 1-3. The impact of each feature is calculated by giving the weights of 6 to the most influential feature (shown as (1)), and 3 and 1 to the second two influential features (i.e (2) and (3)). The impact of each feature is calculated using the weights.

	Wk1	Wk2	Wk3	Wk4	Wk5	Wk6	Wk7	Impact
F1	32	39	49	74**(2)**	76**(2)**	84**(1)**	83**(2)**	**15**
F2	32	39	39	39	39	39	39	
F3	34	45	42	44	45	46	47	
F4	32	39	39	54	34	41	74	
F5	45**(3)**	59	65**(1)**	71**(3)**	75**(3)**	73**(3)**	74	**9**
F6	32	39	39	41	39	39	39	
F7	32	39	39	39	39	39	39	
F8	32	39	39	39	39	46	45	
F9	50**(2)**	61 **(2)**	57	68	70	78**(2)**	77	**9**
F10	58**(1)**	62**(1)**	65 **(1)**	69	71	72	73	**18**
F11	32	39	39	39	39	39	39	
F12	32	39	39	39	39	39	39	
F13	32	39	58**(3)**	82**(1)**	83**(1)**	84**(1)**	85**(1)**	**25**
F14	38	52**(3)**	60**(2)**	71**(3)**	74	78**(2)**	82**(3)**	**9**
F15	32	39	40	40	40	40	40	

1. F13: Periodically log when video is playing
2. F10: View set of shared files
3. F01: Play video
4. F05: Load media
5. F09: View my files
6. F14: Play region and view thread.

The top six features include a combination of *individual* learning activities (e.g. playing a video to watch, as well as viewing the files saved by the student themselves) and *social* learning activities (e.g. periodically making notes and logging information while watching a video, which could be uploaded by the student themselves or their classmates, knowing that the logged items are visible to the rest of the students) all contributing to the learning process. Investigating the above list, one of the interesting observations is that the social learning activity (of interacting with the posted video) has had the largest score (i.e. 25 as shown in Table 4) and is identified as the most important feature.

In the first part of this experiment, the six highest impact features shown before are selected as input to the system and results are demonstrated in Table 5. While the results are comparable to the previous experiment when all the features where used, the outcome exhibits a slight reduction in the prediction accuracy which could be due to some of the conflicting nature of the features (e.g. combining features which are as diverse as having the impact of 25 and 9). Please note that this hypothesis should be treated with caution as a more in-depth analysis is required to verify this thought. In the second experiment of this section (and in an attempt to explore the previous hypothesis), only two of most significant features (which are the social learning features) are used; the two features used are F13 (periodically log when video is playing) and F10 (view set of shared files). As shown in Table 6, the results demonstrate the highest prediction

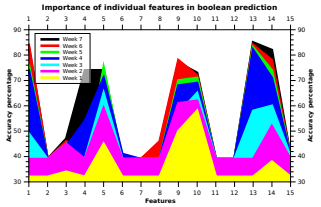


Figure 3: Impact of using individual features. Layers in this diagram represents accuracy of features in each week.

accuracy found on this dataset from week 4 of the term. The median prediction accuracy for week 4 is 83% which is 10% and 9% higher than when six most important features and all features are used respectively (see Tables 3 and 5). Comparing the prediction accuracy reported in Tables 3, 5 and 6 shows that while using the two most important social features, does not improve the prediction accuracy at the very early stages of the term (week 1, 2 and 3), it does enable a stronger prediction from the middle week (week 4) onwards. While this may or may not be extendible to other case studies, this finding highlights the usefulness of investigating the positive or negative nature of social features in online learning environment.

6.4 Discussion

Here, the key research questions raised in Section 1 are discussed next and various aspects of the findings are analysed. As stated in the first research question, this paper applies the Stochastic Diffusion Search (SDS) to classify educational data. The potential and strength of the this algorithm is demonstrated in the results and the flexibility of the algorithm to deal with various feature vector is also highlighted. Given SDS's existing 'partial function evaluation' feature (i.e. each micro-feature, or attribute, is used independently of the others in the test phase), and the resulting low computational cost of comparing samples, this algorithm is likely to be particularly useful when applied to problems with huge dimensionality, which is usually the case in educational data analysis. In this context, the link between cheap computational cost and scalability is the subject of an ongoing research. To address the second research question, three experiments are run (see Fig. 4). Neither of the three experiments (using all features, 6 best features, and 2 best social features) are able to provide a reliable prediction in the first three weeks (e.g. less than 60%) of this seven-week course analysed in this paper; it is worth noting that in the first three weeks, when the social features are solely used in the analysis, the algorithm exhibits the worst outcome, possibly due to the lack or reduced social interactions among the students in the very first a few weeks. However, looking at the performance of the algorithm in weeks 4-7, it can be seen that while using all features or the six most significant features are not causing a huge difference in week 4, the gap widens from week 5-7, showing that the use of all features could prove better than the top six features. On the other hand, having picked the two top features (which are inherently social in nature and involve interactions with other students), the algorithm outperforms the other configurations and provides the prediction accuracy as high as 83% in week 4, and up to nearly 90% in week 7. To address the third research question, the role of social features reflecting the social learning activities are investigated. These features are shown to have played a significant role and as highlighted in the fourth research question, identifying the link between the *social* learning activities and the *student success* in this dataset could give insight to course developers and educators with regards to designing and delivering

Table 5: Combining the most influential six features.

	Mean	Median	StDev	Min	Max
Week 1	45.2	45.5	4.41	32	52
Week 2	52.5	52	2.21	48	57
Week 3	59.57	60	2.75	46	63
Week 4	72.67	74	6.22	62	82
Week 5	72.67	75	7.84	57	83
Week 6	78.43	82	8.03	55	86
Week 7	79.77	80.5	4.85	68	87

Table 6: Combining two of the most influential features.

	Mean	Median	StDev	Min	Max
Week 1	32	32	0	32	32
Week 2	39	39	0	39	39
Week 3	54.37	54	1.03	52	57
Week 4	81.4	83	4.00	66	84
Week 5	81.77	82.5	2.42	75	85
Week 6	87.4	88	1.00	85	89
Week 7	87.8	88	0.76	86	89

course activities. Having established a link between social learning and student success, the results highlight the possibility of providing a more surgical feedback (based on the important features verses all features) to the students who are picked as likely to fail by the system. This study has also shown the importance of the social features used which could be of help when providing feedback to students.

7. CONCLUSIONS

The paper demonstrates the ability of the proposed swarm intelligence classifier in dealing with the existing educational data. The simplicity of this algorithm with one tunable parameter (i.e. agent size) makes it an attractive technique to use. One of the key contribution of the paper is to provide evidence that the data collected on our social learning platform (delivered to several thousand students on Coursera), which records the way in which students share, view and comment on each other's work, is related to performance. Specifically, whilst predicting the final fail/pass of students might be difficult on the first few weeks, the prediction accuracy rises to 83% in week 4 and as high as 89% on week 7. Given two of the social features are demonstrated to have played an important role in the prediction accuracy of the algorithm, as the work progresses, the authors will start to look at questions such as what social behaviours are the best predictors of performance? When can such predictions be made? What kinds of social behaviour impact upon the predicted grades of students? Is it possible to help design interventions for students and tutors to help each other? Finally, after several years of building a system through participatory design and concentrating on the user experience, we are now in a position to use a data driven approach to build systems to support communities of learners.

8. REFERENCES

[1] E. Aguiar, H. Lakkaraju, N. Bhanpuri, D. Miller, B. Yuhas, and K. L. Addison. Who, when, and why: a machine learning approach to prioritizing students at risk of not graduating high school on time. *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 93–102, 2015.

[2] M. M. al-Rifaie and M. Bishop. Stochastic diffusion search review. In *Paladyn, Journal of Behavioral Robotics*, volume 4(3), pages 155–173. Paladyn, Journal of Behavioral Robotics, 2013.

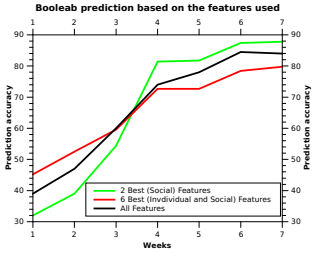


Figure 4: Impact of using various features in the accuracy of the prediction

[3] C. Brooks, C. Thompson, and S. Teasley. Who you are or what you do: Comparing the predictive power of demographics vs. activity patterns in massive open online courses (moocs). In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 245–248. ACM, 2015.

[4] C. a. Coleman, D. T. Seaton, and I. Chuang. Probabilistic Use Cases: Discovering Behavioral Patterns for Predicting Certification. *Proceedings of the Second (2015) ACM Conference on Learning @ Scale - L@S '15*, pages 141–148, 2015.

[5] N. M. Dowell, O. Skrypnyk, S. Joksimović, A. Graesser, S. Dawson, D. Gašević, P. de Vries, T. Hennis, and V. Kovanović. Modeling learnersâĂŹ social centrality and performance through language and discourse. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[6] A. Elbadrawy, R. S. Studham, and G. Karypis. Collaborative multi-regression models for predicting students' performance in course activities. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge - LAK '15*, pages 103–107, 2015.

[7] Y. Gong and J. E. Beck. Towards detecting wheel-spinning: Future failure in mastery learning. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 67–74. ACM, 2015.

[8] L. P. Macfadyen and S. Dawson. Mining LMS data to develop an "early warning system" for educators: A proof of concept. *Computers and Education*, 54(2):588–599, 2010.

[9] J. K. Olsen, V. Aleven, and N. Rummel. Predicting student performance in a collaborative learning environment. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[10] C. Reas and B. Fry. *Processing: A Programming Handbook for Visual Designers and Artists*. The MIT Press, aug 2007.

[11] T. Rogers, C. Colvin, and B. Chiera. Modest analytics: using the index method to identify students at risk of failure. In *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge*, pages 118–122. ACM, 2014.

[12] J. Rollinson and E. Brunskill. From predictive models to instructional policies. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[13] J. Whitehill, J. J. Williams, G. Lopez, C. A. Coleman, and J. Reich. Beyond prediction: First steps toward automatic intervention in mooc student stopout. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[14] D. Yang, T. Sinha, D. Adamson, and C. P. Rosé. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-Driven Education Workshop*, 2013.

[15] C. Ye, J. S. Kinnebrew, G. Biswas, B. J. Evans, D. H. Fisher, G. Narasimham, and K. A. Brady. Behavior prediction in moocs using higher granularity temporal information. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 335–338. ACM, 2015.

Predicting Student Progress from Peer-Assessment Data

Michael Mogessie Ashenafi
University of Trento
via Sommarive 9, 38123
Trento, Italy
+390461285251
michael.mogessie@unitn.it

Marco Ronchetti
University of Trento
via Sommarive 9, 38123
Trento, Italy
+390461282033
marco.ronchetti@unitn.it

Giuseppe Riccardi
University of Trento
via Sommarive 9, 38123
Trento, Italy
+390461282087
giuseppe.riccardi@unitn.it

ABSTRACT
Predicting overall student performance and monitoring progress have attracted more attention in the past five years than before. Demographic data, high school grades and test result constitute much of the data used for building prediction models. This study demonstrates how data from a peer-assessment environment can be used to build student progress prediction models. The possibility for automating tasks, coupled with minimal teacher intervention, make peer-assessment an efficient platform for gathering student activity data in a continuous manner. The performances of the prediction models are comparable with those trained using other educational data. Considering the fact that the student performance data do not include any teacher assessments, the results are more than encouraging and shall convince the reader that peer-assessment has yet another advantage to offer in the realm of automated student progress monitoring and supervision.

Keywords
Progress prediction; peer-assessment; learning analytics.

1. INTRODUCTION
Common examples of traditional student assessment methods are end-of-course examinations that constitute a very high proportion of final scores and other standardised and high stakes tests.

There are, however, other student-centric, yet less practiced, forms of assessment. Formative assessment is a fitting example [7]. It is designed with the goal of helping students meet specified learning goals through continuous discussion, gauging and reporting of their performance.

Peer-assessment is another form of assessment, which may be designed with summative or formative goals. It is a form of assessment where students evaluate the academic products of their peers [15].

Automated peer-assessment provides a rich platform for gathering data that can be used to monitor student progress. In such context, another dimension of peer-assessment emerges – its potential to serve as a foundation for building prediction models on top of.

In this study, we demonstrate how this potential can be exploited by building linear regression models for predicting students' weekly progress and overall performance for two undergraduate-level computer science courses that utilised an automated peer-assessment.

The rest of this paper is organised as follows. The next section discusses recent advances in student performance prediction. Section 3 presents a brief overview of the web-based peer-assessment platform using which the data was collected. Section 4 discusses details of the data and the features that were selected to build the prediction models. Section 5 provides two interpretations of student progress and details how these interpretations determine which data shall be used for building the models. Section 6 introduces the reader to how the prediction models are trained and provides details of the prediction performance evaluation metrics reported. Section 7 discusses the first interpretation of progress prediction and demonstrate the respective prediction models. Section 8 builds upon the second interpretation and follows the same procedure as section 7. Section 9 provides a short discussion and conclusion of the study.

2. PREVIOUS WORK IN PREDICTING STUDENT PERFORMANCE
Earlier studies in student performance prediction investigated the correlation between high school grades and student demographic data and success in college education as evidenced by successful completion of studies [1, 6].

Unsurprisingly, many of these studies were conducted by scholars in the social sciences and involved the use of common correlation investigation methods such as linear and logistic regression. The large majority of recent studies have, however, been conducted in the computer science discipline. These studies use data from courses administered as part of either computer science or engineering programmes at the undergraduate level. Of these, many focus on predicting performance of freshman and second year students enrolled in introductory level courses.

A generic approach to student performance prediction is to predict overall outcome such as passing or failing a course or even forecasting successful completion of college as marked by graduation [9, 13, 14]. A further step in such an approach may include predicting the classification of the degree or achievement [8].

More fine-grained and sophisticated approaches involve predicting actual scores for tests and assignments as well as final scores and grades for an entire course.

Due to the varying nature of the courses and classes in which such experiments are conducted and advanced machine learning techniques that are readily available as parts of scientific software packages, the number distinct, yet comparable, studies in performance prediction has been growing steadily. Another factor, the proliferation of MOOCs, has fuelled this growth with the immense amount of student activity data generated by these platforms.

Examples of studies that utilise information from students' activities in online learning and assessment platforms in predicting performance include [2, 10, 11].

Apart from predicting end-of-course or end-of-programme performance, prediction models may be used to provide continuous predictions that help monitor student progress. When used in this manner, such prediction models could serve as instruments for early detection of at-risk students. Information provided by these models could then serve the formative needs of both students and teachers. Studies that demonstrate how prediction models can be used to provide continuous predictions and may serve as tools of early intervention include [5, 10].

The most common algorithms in recent literature that are used for making performance predictions are Linear Regression, Neural Networks, Support Vector Machines, Naïve Bayes Classifier, and Decision Trees.

Studies that follow less common approaches include those that use smartphone data to investigate the correlation between students' social and study behaviour and academic performance [16] and those that perform Sentiment Analysis of discussion form posts in MOOCs [4].

Two studies that present algorithms developed for the sole purpose of student performance prediction are [12] and [17].

3. THE PEER-ASSESSMENT PLATFORM

In 2012, an experimental web-based peer-assessment system was introduced into a number of undergraduate level courses at an Italian university. Using this peer-assessment system, students completed three sets of tasks during each week of the course. The weekly cycle started with students using the online platform to submit questions about topics that were recently discussed in class. These questions were then reviewed by the teacher, who would select a subset and assign them to students, asking them to provide answers. The assignment of the questions to students was automatically randomised by the system, which guaranteed anonymity of both students who asked the questions and those who answered them. Once this task was completed, the teacher would assign students the last task of the cycle, in which they would rate the answers provided by their peers and evaluate the questions in terms of their perceived difficulty, relevance and interestingness.

Eight cycles of peer-assessment were carried out in two undergraduate-level computer science courses, IG1 and PR2. Participation in peer-assessment activities was not mandatory. However, an effort to engage students in these tasks was made by awarding students with bonus points at the end of the course in accordance with their level of participation and the total number of peer-assigned marks they had earned for their answers. The design and development of the peer-assessment platform and the theoretical motivations for it are discussed in [3].

4. THE DATA

Because participation in peer-assessment tasks was not mandatory, there was an apparent decline in the number of participants towards the end of both courses. In order to minimise noise in the resulting prediction models, only peer-assessment activity data of those students who completed at least a third of the total number of tasks and for whom final grades were available were selected for building the models. This led to the inclusion of 115 student records for IG1 and 114 for PR2.

In a previous study [2], a linear regression model for predicting final scores of students using the same data was discussed. Experiments in that study revealed that predicting the range within which a score would fall was more accurate than predicting actual scores. Indeed, this is tantamount to predicting grades. During the experiments in that study, although attempts were made to build classification models that predicted grades in a multiclass classification manner, the results were found to be much better when actual scores were predicted using linear regression and those scores were mapped to grades according to mappings which were specified beforehand. Hence, the authors decided to apply those techniques in this study as well.

Grades are arguably the ideal approach to judging the performance levels of students because they usually span a wider range of scores, within which a student's scores are likely to fall if the student sits the same exam in relatively quick successions. Consequently, scores predicted by the linear regression models were transformed into grades.

The parameters used to build the linear regression models are:

Tasks Assigned (TA) – The number of tasks that were assigned to the student

Tasks Completed (TC) – The number of tasks that the student completed

Questions Asked (QAS) – The number of 'Ask a Question' tasks the student completed

Questions Answered (QAN) – The number of 'Answer a Question' tasks the student completed

Votes Cast (VC) – The number of 'Rate Answers' tasks the student completed

Questions picked for answering (QP) – The number of the student's questions that were selected by the teacher to be used in 'Answer a Question' tasks

Votes Earned (VE) – The number of votes the student earned for their answers

Votes Earned Total Difficulty (VED) – The sum of the products of the votes earned for an answer and the difficulty level of the question, as rated by students themselves, for all answers submitted by the student

Votes Earned Total Relevance (VER) – The sum of the products of the votes earned for an answer and the relevance level of the question, as rated by students themselves, for all answers submitted by the student

Votes Earned Total Interestingness (VEI) – The sum of the products of the votes earned for an answer and the interestingness level of the question, as rated by students themselves, for all answers submitted by the student

Selected Q total difficulty (SQD) – The sum of the difficulty levels of the student's questions, as rated by students themselves, which were selected to be used in subsequent tasks

Selected Q total relevance (SQR) – The sum of the relevance levels of the student's questions, as rated by students themselves, which were selected to be used in subsequent tasks

Selected Q total interestingness (SQI) – The sum of the interestingness levels of the student's questions, as rated by students themselves, which were selected to be used in subsequent tasks

Details of the linear regression model, possible justifications for its prediction errors and experiments comparing its performance to baseline predictors are provided in [2].

5. TWO INTERPRETATIONS OF PROGRESS PREDICTION

Monitoring student progress using prediction models requires making predictions using evolving student data at several intervals. Continuous peer-assessment data are the ideal candidate for building such prediction models.

Through years of experience, teachers are usually able to make educated guesses about how student are likely to perform at end-of-course exams by studying their activities throughout the course. Prediction models that use data from previous editions of the same course adopt and formalise such experience with greater efficacy.

Indeed, prediction models can be used not only to make one-off predictions of student performance at the end of a course, but also at several intervals throughout the course. While continuous predictions focus on determining student progress by evaluating performance at different stages, one-off predictions put more importance on whether a student would finally pass a course on not.

This study focuses on the former, making continuous predictions to measure student progress and provides two interpretations of student *progress*.

One interpretation compares a student's standing at any point in the course to the standings of students at the same point but from previous editions of the course. For instance, in a previous edition of a course, if student performance data at every week of the course were collected and if these data were complemented with end-of-course grades, in subsequent editions of the course, a student's performance at any week would be compared to the performances of students at that specific week in the previous edition of the course and the respective grade for the student's level of performance could be predicted. In favour of brevity, this interpretation of progress will be referred to as ***Progress Type A***.

The other interpretation focuses on evaluating how far a student is from achieving goals that they are expected to achieve at the end of a course. In a fairly simplified manner, this evaluation may be made by comparing the expected final grade of student at any point during the course to what is deemed to be a desirable outcome at the end of the course. For instance, predicting a student's end-of-course grade in the second week of an eight-week course and comparing that predicted grade to what is considered to be a favourable grade at the end of the course, which is usually in the range A+ to B-, can provide information about how far the student is from achieving goals that are set out at the beginning of the course. In favour of brevity, this interpretation of progress will be referred to as ***Progress Type B***.

6. TRAINING AND MEASURING THE PERFORMANCE OF THE PREDICTION MODELS

Peer-assessment data collected during the course were divided into weekly data according to the three sets of tasks completed every week. The final score of each student for the course was then converted into one of four letter grades.

The data for each week incorporate the data from all previous weeks. In this manner, the prediction model for any one week is built using more performance data than its predecessors. Naturally, the data used to build the model for the first week would be modest and the data for the final week model would be complete. In general, the performances of models from consecutive weeks were expected to be better.

A common metric used in measuring the performance of linear regression prediction models is the Root Mean Squared Error (RMSE). While RMSE provides information about the average error of the model in making predictions, the conversion of numerical scores to letter grades enables using more informative performance evaluation metrics.

The conversion of numerical scores to letter grades transforms this prediction into a classification problem, with grades treated as class labels. Although multiclass classification algorithms were not applied due to their relatively low performance for this specific task, transformation of predicted scores into grades permitted the application of any of the classification performance evaluation metrics. Therefore, performance is reported in terms of precision, recall, F1, False Positive Rates (FPR) and True Negative Rates (TNR).

When evaluating student performance prediction models, the two questions that are more critical than others are:

- How many of the students the model predicted not to be at-risk were actually at-risk and eventually performed poorly (False Positives) and
- How many of the students that the model predicted to be at-risk of failing were indeed at-risk (True Negatives).

A prediction model with a high FPR largely fails to identify students who are at risk of failing. Conversely, a model with a high TNR identifies the majority of at-risk students. The ideal prediction model would have a very low FPR and, consequently, a very high TNR.

The prediction models are evaluated at two levels. The first level is their performance in making exact prediction of grades. The second is their performance in making a prediction that is within a one grade-point range of the actual grade.

For the purpose of this study, the performance metrics are defined as follows.

Grade – Any of the letters A, B, C, D – A and B denote high performance levels and C and D, otherwise. Although C is usually a pass grade, it is generally not favourable and considered to be a low grade.

Positive – A prediction that is either A or B

Negative – A prediction that is either C or D

True – A prediction that is either the exact outcome or falls within a one grade-point range of the actual outcome

False – A prediction that is not True

Any combination of positive or negative predictions with true or false predictions yields one of the following counts – True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

Important statistics that use these counts are Precision (P), Recall (R) and, inherently, F1 scores.

It should be noted that FPR and TNR provide two interpretations of the same outcome and that they are inversely proportional. Indeed, FPR = 1 – TNR.

7. MODELLING PROGRESS TYPE A

This type of progress monitoring compares a student's current progress at any week during the course to the progresses of past

students at the same week of the course. The question that such an approach aims to answer is: 'Compared to how other students were doing at this stage in the past, how well is this student doing now?' 'How well' the student is doing is predicted as follows. First, a linear regression model is built using data collected from the first week to the week of interest. This data comes from a previous edition of the course and the predicted variable is the final score or grade, which is already available. Then, the student's performance at the week in question, represented using the parameters in section 4, is fed to the model to make a prediction. Such weekly information shall provide insight into whether the student is likely to fall behind other students or not.

The prediction errors for the course PR2 gradually decreased for successive weeks, as expected. For IG1, however, early decreases were followed by increases and a slight decrease in the final week. The average RMSE for PR2 for the eight models was 3.4 while it was 3.6 for IG1. The scores predicted were in the range 18 to 30 Figure 1 shows the weekly prediction errors for each course.

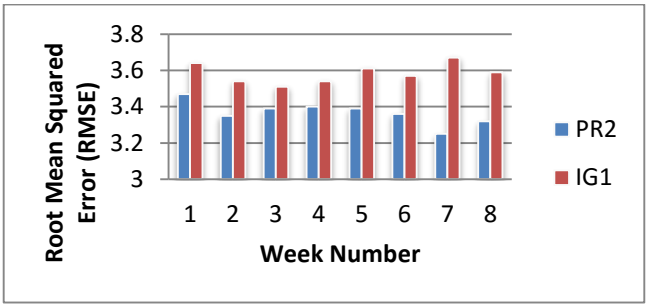


Figure 1. Prediction Errors for the models of each course over eight weeks

Low performance levels were recorded for exact grade prediction of the models for both courses. Specifically, High false positive rates persisted throughout the eight-week period.

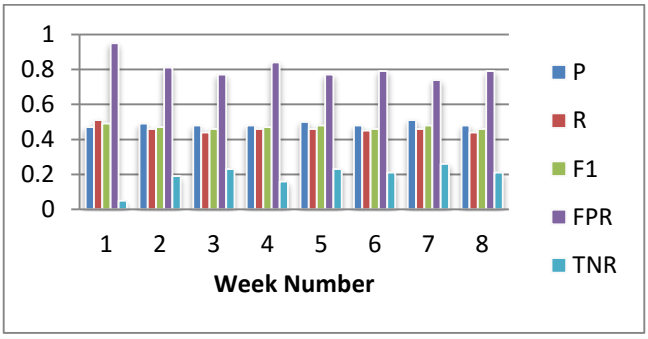


Figure 2. Exact grade prediction performance for PR2

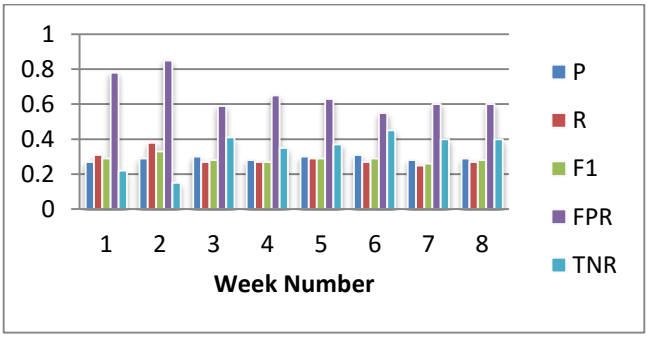


Figure 3. Exact grade prediction performance for IG1

As expected, performance levels of the models for both courses significantly increased for within one grade-point predictions. Low FPR and, consequently, high TNR were recorded even in the first week and performance increased gradually for both courses over the eight-week period.

The models that made within-one-grade-point predictions performed well from the very first week of the course. Although predictions are not made on exact grades, the wider range helps lower the rate of false positives and increase true positives. The same consideration may lead to an increase in false negatives, and hence, a decrease in true positives. However, the high precision and recall values for these models attest that this is not so in this case.

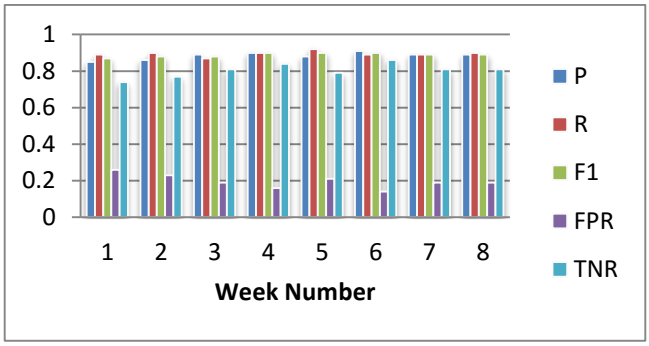


Figure 4. Within-one-grade-point prediction performance for PR2

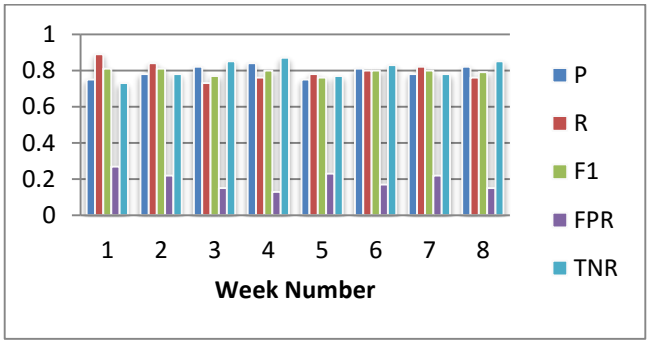


Figure 5. Within-one-grade-point prediction performance for IG1

8. MODELLING PROGRESS TYPE B

The focus of this type of measuring progress can be informally described as *measuring the gap* between a student's performance *now* and what it is expected to be *at the end* of the course. Modelling this type of progress only requires building a single linear regression model using the entire data from previous editions of the same course. Then, a student's performance data at any week, which is represented by an instance of the values for the parameters discussed in section 4, is fed to the linear regression equation to compute the expected score of the student. This score is then transformed to a grade. Such weekly information would help keep track of a student's progress towards closing this gap and achieving the desired goals.

The prediction errors of this model for the eight weeks are reported in Figure 6. The prediction errors for both courses were significantly lower than those for Progress Type A, with the model for PR2 having an average RMSE of 3.0 and the model for IG1 scoring a higher average RMSE of 3.5. Moreover, prediction errors for both courses consistently decreased throughout the eight weeks.

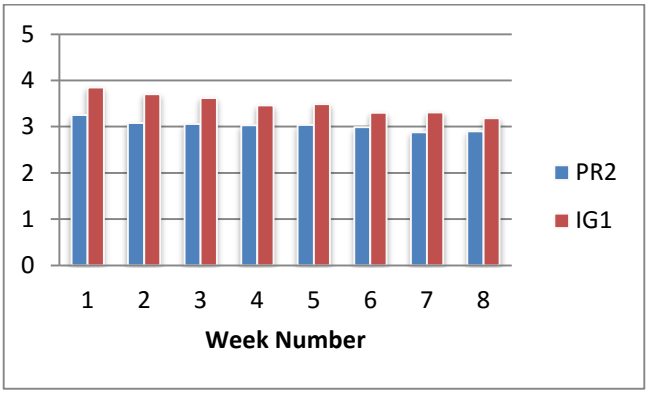


Figure 6. Prediction errors of the model of the two courses over an eight-week period

Exact grade prediction performance, although better than that of Progress Type A, was still low for both courses.

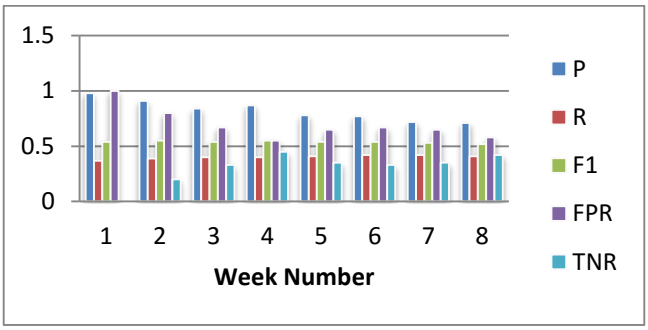


Figure 7. Exact grade prediction performance for PR2

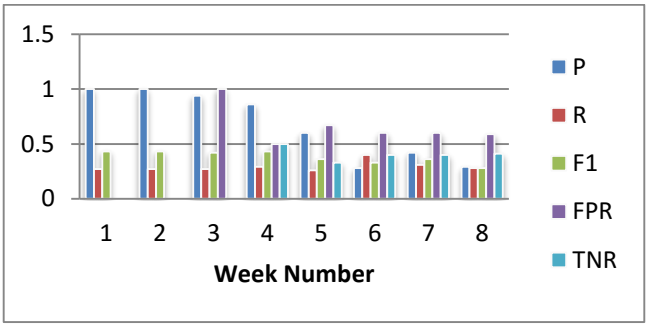


Figure 8. Exact grade prediction performance for IG1

Similar to the models of Progress Type A, this model had very high levels of performance in predicting grades that fell within one grade-point of the actual grades. Prediction performance was very high in the first week and consistently increased, albeit by small amounts, throughout the remaining weeks for both courses.

Missing FPR and TNR values for both courses in the beginning weeks imply that predictions of the model were distributed over TP and FN values. However, high precision values during those weeks indicate that FN values were very low.

Overall, the model for Progress Type B outperformed the models that from Progress Type B, for both courses.

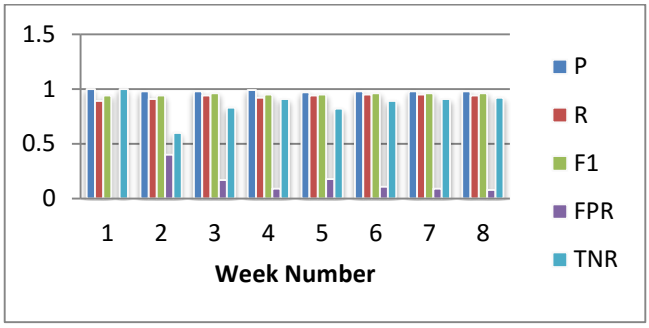


Figure 9. Within-one-grade-point prediction performance for PR2

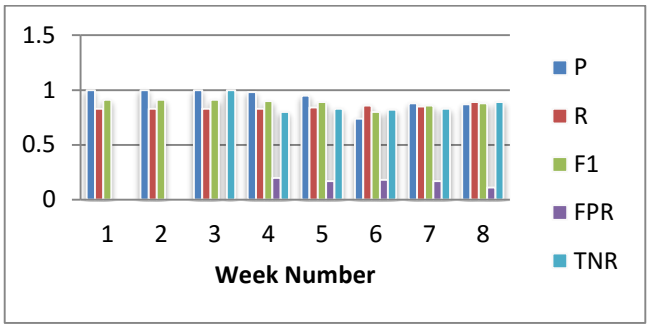


Figure 10. Within-one-grade-point prediction performance for IG1

9. DISCUSSION AND CONCLUSION

From peer-assessment tasks that were conducted over an eight-week period in two courses, data were used to build several prediction models according to two distinct interpretation of performance prediction. While the first interpretation focused on comparing the performance of a student at any week during the course to those of past students' performance levels obtained in the same week, the second focused on measuring how far a student is from achieving the desired level of performance at the end of a course.

The approach of using data from previous editions of the same course may raise doubts as to whether different editions of the same course are necessarily comparable. However, the extents to which the prediction models discussed here performed should convince the reader that this is indeed possible. Performance of the models is in fact expected to improve with increase in the number of previous editions of the course used as input for making predictions. Indeed, the long-term consistency in the number of below-average, average and above average students over many editions of a course is how many teachers usually measure the overall difficulty level of questions that they include in exams.

Although exact grade predictions did not produce satisfactory levels of performances for either approach, high levels of performance were obtained for both interpretations of student progress when making within-one-grade-point predictions. This signifies the promising potential of carefully designed peer-assessment and the prediction models built using data generated from it as tools of early intervention.

While the statement that a student's performance at the end of a course can be fairly predicted as early as the first weeks of the course from their peer-assessment activity may be construed as simplistic, it is worth noting that the experiments were carried out

in two computer science courses and that the results suggest otherwise.

While a comparison between the performances of the models for the two courses may be made, the reasons behind one model outperforming the other may be latent at this stage and require detailed investigation. Hence, the authors decided to defer making such comparisons until a later stage.

10. REFERENCES

[1] Al-Hammadi, A. S., and Milne, R. H. (2004). A neuro-fuzzy classification approach to the assessment of student performance. In Fuzzy Systems, 2004. Proceedings. 2004 IEEE International Conference on (Vol. 2, pp. 837–841 vol.2). http://doi.org/10.1109/FUZZY.2004.1375511

[2] Ashenafi, M. M., Riccardi, G., and Ronchetti, M. (2015). Predicting students' final exam scores from their course activities. In Frontiers in Education Conference (FIE), 2015 IEEE (pp. 1–9). http://doi.org/10.1109/FIE.2015.7344081

[3] Ashenafi, M.M., Riccardi, G., & Ronchetti, M. (2014, June). A Web-Based Peer Interaction Framework for Improved Assessment and Supervision of Students. In World Conference on Educational Multimedia, Hypermedia and Telecommunications (Vol. 2014, No. 1, pp. 1371-1380).

[4] Chaplot, D. S., Rhim, E., and Kim, J. (2015). Predicting student attrition in moocs using sentiment analysis and neural networks. In Proceedings of AIED 2015 Fourth Workshop on Intelligent Support for Learning in Groups.

[5] Coleman, C. A., Seaton, D. T., and Chuang, I. (2015). Probabilistic Use Cases: Discovering Behavioral Patterns for Predicting Certification. In Proceedings of the Second (2015) ACM Conference on Learning @ Scale (pp. 141–148). New York, NY, USA: ACM. http://doi.org/10.1145/2724660.2724662

[6] Evans, G. E., and Simkin, M. G. (1989). What best predicts computer proficiency?. Communications of the ACM, 32(11), 1322-1327. http://doi.org/10.1145/68814.68817

[7] Harlen, W., & James, M. (1997). Assessment and Learning: differences and relationships between formative and summative assessment. Assessment in Education, 4(3), 365-379. http://doi.org/10.1080/0969594970040304

[8] Jiang, S., Williams, A., Schenke, K., Warschauer, M., and O'dowd, D. (2014). Predicting MOOC performance with week 1 behavior. In Educational Data Mining 2014.

[9] Karamouzis, S. T., and Vrettos, A. (2009). Sensitivity Analysis of Neural Network Parameters for Identifying the Factors for College Student Success. In Computer Science and Information Engineering, 2009 WRI World Congress on (Vol. 5, pp. 671–675). http://doi.org/10.1109/CSIE.2009.592

[10] Koprinska, I., Stretton, J., and Yacef, K. (2015). Predicting Student Performance from Multiple Data Sources. In C. Conati, N. Heffernan, A. Mitrovic, and M. F. Verdejo (Eds.), Artificial Intelligence in Education SE - 90 (Vol. 9112, pp. 678–681). Springer International Publishing. http://doi.org/10.1007/978-3-319-19773-9_90

[11] Manhães, L. M. B., da Cruz, S. M. S., and Zimbrão, G. (2014). WAVE: An Architecture for Predicting Dropout in Undergraduate Courses Using EDM. In Proceedings of the 29th Annual ACM Symposium on Applied Computing (pp. 243–247). New York, NY, USA: ACM. http://doi.org/10.1145/2554850.2555135

[12] Meier, Y., Xu, J., Atan, O., and van der Schaar, M. (2015). Predicting Grades. Signal Processing, IEEE Transactions on, PP (99), 1. http://doi.org/10.1109/TSP.2015.2496278

[13] Nghe, N. T., Janecek, P., and Haddawy, P. (2007). A comparative analysis of techniques for predicting academic performance. In Frontiers in Education Conference - Global Engineering: Knowledge without Borders, Opportunities without Passports, 2007. FIE '07. 37th Annual (pp. T2G–7–T2G–12). http://doi.org/10.1109/FIE.2007.4417993

[14] Plagge, M. (2013). Using Artificial Neural Networks to Predict First-year Traditional Students Second Year Retention Rates. In Proceedings of the 51st ACM Southeast Conference (pp. 17:1–17:5). New York, NY, USA: ACM. http://doi.org/10.1145/2498328.2500061

[15] Topping, K.J. (1998). Peer Assessment Between Students in Colleges and Universities. Review of educational Research, 68(3), 249-276. http://doi.org/10.3102/00346543068003249

[16] Wang, R., Harari, G., Hao, P., Zhou, X., and Campbell, A. T. (2015). SmartGPA: How Smartphones Can Assess and Predict Academic Performance of College Students. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (pp. 295–306). New York, NY, USA: ACM. http://doi.org/10.1145/2750858.2804251

[17] Watson, C., Li, F. W. B., and Godwin, J. L. (2013). Predicting Performance in an Introductory Programming Course by Logging and Analyzing Student Programming Behavior. In Advanced Learning Technologies (ICALT), 2013 IEEE 13th International Conference on (pp. 319–323). http://doi.org/10.1109/ICALT.2013.99

Topic-wise Classification of MOOC Discussions: A Visual Analytics Approach

Thushari Atapattu, Katrina Falkner, Hamid Tarmazdi
School of Computer Science
University of Adelaide
Adelaide, Australia
{firstname.lastname}@adelaide.edu.au

ABSTRACT

With a goal of better understanding the online discourse within the Massive Open Online Course (MOOC) context, this paper presents an open source visualisation dashboard developed to identify and classify emergent discussion topics (or themes). As an extension to the authors' previous work in identifying key topics from MOOC discussion contents, this work visualises lecture-related discussions as a graph of relationships between topics and threads. We demonstrate the visualisation using three popular MOOCs offered during 2013. This work facilitates the course staff to locate and navigate the most influential topic clusters as well as the discussions that require intervention by connecting the topics with the corresponding weekly lectures. Further, we demonstrate how our interactive visualisation can be used to explore correlation between discussion topics and other variables such as views, posts, votes, and instructor intervention.

Keywords

Visualisation, learning analytics, topic model, MOOC, online discourse, discussion forum.

1. INTRODUCTION

Within the educational context, visualisation of learning analytics, often known as 'visual analytics', provides insights for many end users including teachers, learners, researchers, educational platform developers, and institutions. According to Thomas and Cook [1], visual analytics focuses on analytical reasoning facilitated by interactive visualisation interfaces. In the educational context, visual analytics support teachers in identifying at-risk students, analysing students' engagement and performance of the course, social collaborations, and developing analytics on the students' online discourse. Visualisation dashboards also support self-evaluation for students in reflecting on their own learning process, setting goals and monitoring progress to achieve these goals.

Visual analytics are often useful in large to massive classrooms such as Massive Open Online Courses (MOOCs), facilitating the understanding of interesting patterns in large volume of students' data, which is challenging to observe using statistical analysis. Visualising the patterns of student engagement (e.g. lecture/forum view), behavior, social interactions and their relationship with final grade/performance has being a focus of many studies [2-4].

Even though the *system-generated* analytics on students' engagement and behavior are important to identify patterns that positively correlate with the successful learning outcomes or attrition, it is likely that these can generate some inconsistencies. For instance, a download of a lecture does not necessarily imply student engagement. Similarly, it is uncertain whether an up-vote of a forum post means the learner has an interest in the content or, alternatively, that they have problems associated with the topic discussed in the post. Therefore, the analysis of *learner-generated* online discourse (i.e. content) facilitates the interpretation of learners' cognitive processes as well as situating learner behavior in context. According to Mercer [5], the sociocultural perspective highlights "the possibility that educational success and failure may be explained by the quality of educational dialogue, rather than simply in terms of the capability of individual students or the skill of their teachers". This includes identification of individual's understanding of – and interest in – particular course content, and their level of expertise and activity in seeking assistance to rectify conflicts, provide opinions and interact with instructors and peers through dialogs [6, 7]. Existing research focuses on visualising discussion participation and social interactions [8, 9], however, analysis and the visualisation of discussion content (i.e. written discourse) is lacking. Furthermore, there is no support from existing MOOC models to effectively organise and visualise these data. In a preliminary work, Chen [10] and Speck et al. [11] focus on identifying and visualising topic models from online discussion platforms.

Due to the overwhelming abundance of information generated within MOOCs, it is challenging for the learners and the course staff to effectively locate and navigate information. Therefore, topic analysis from MOOC discussions is important in identifying main themes from students' discussions, supporting forum facilitators to become aware of the key themes and the amount of discussions in each theme. We have previously developed a framework for discourse analysis in the MOOC context that identifies latent discussion topics [12]. Our work connects lecture-related discussion topics with the corresponding weekly lectures, allowing course staff to visualise the discussions as clusters of lectures. We have experimented with our framework using three MOOCs and obtained promising results [12].

This paper focuses on developing an open source dashboard to visualise topics extracted from MOOC discussion contents. Our topic visualisation dashboard expects to answer two main questions important to the educators: *What are the emergent topics?,* and *What topics need more attention?*. Further, we also explore the topic distribution using additional variables such as views, votes, replies, and the degree of instructor intervention and answer the questions including '*what is the relationship between topics and views?*', '*what is the relationship between topics and votes*', and '*what is the relationship between topics and instructor replies*'. These questions have emerged from the authors' involvement in several MOOC courses and environments to explore key course management issues and pedagogical decisions. To answer these questions, we conducted

a statistical analysis using 3 popular MOOCs – *Machine Learning*, *Statistics* and *Psychology* and compared the results using the proposed visualisation dashboard.

2. BACKGROUND

Visual analytics within the educational context often facilitate educators in understanding large amount of learners' data to make inferences. Learners' data can be categorised as *system-generated* and *learner-generated*. System-generated data (also known as clickstream data) are generally analysed and visualised to predict the performance (e.g. CourseSignals [4]). Social Networks Adapting Pedagogical Practice (SNAPP) [8] visualises the evolution of social interactions among participants of online discussion forums.

Within the MOOC context, Coffrin et al. [2] visualises patterns of engagement and performance based on student types (e.g. auditor, active, qualified). Xu et al. [13] utilises visual analytics to explore the correlation between student behavior and student success. In a preliminary work, they analysed five MOOCs using a commercial visualisation software called *Tableau* and reported that there are multiple ways to be successful in a course (e.g. submitting quizzes, lecture views). While there is considerable, as highlighted above, contributing to the development of visual analytics capacity to better understand system-generated educational data, visualisation systems to understand learner-generated data (e.g. online discourse) is lacking.

ForumDash, a preliminary work by Speck et al. [11], focuses on visualising which students are contributing, struggling, or distracted in order to facilitate instructors in targeting their efforts effectively. Using three visualisation tools, ForumDash attempts to provide insights for teachers on which students contribute to most discussions (i.e. Thought-leaders), identify topic clusters to determine the popular topics, and through a 'contribution score visualisation', students' are capable of monitoring how much they are contributing to discussion forums compared to their peers. KISSME (The Knowledge, Interaction and Semantic Student Model Explorer) is a visualisation framework to analyse online discourse with the aim of understanding the nature of interactions among learners including contributions and relationships using LSA and social network analysis [14]. Chen [10] conducts a preliminary study on visualising topic models from online discussion platforms. Another existing tool of interest that takes elements of topic identification and social network analysis is 'Cohere' [15]. The authors use argument-mapping techniques to analyse the discussion posts based on some dimensions such as whether the post is an idea, question, or opinion, in measuring the learner's performance and attention. Topic Facet Model (TFM) incorporates forum posts (mainly questions) about Java from StackOverflow for topic analysis and visualisation [16].

Thus, our motivation for developing this research occurs due to a lack of an established research to produce 'labeled' topic models to analyse overwhelming abundance of MOOC discussion contents and visualisations.

3. TOPIC VISUALISATION DASHBOARD

The overview of topic analysis and visualisation is shown in the Figure 1. The process of topic analysis is briefly discussed in Section 3.1 and the full description can be found in the authors' previous works [12] (full analysis of this work is under review).

3.1 Topic Analysis

Our previous work focuses on identifying topic clusters from *lecture-related* MOOC discussion contents. For this, we have used a state of the art topic modeling technique called Latent Dirichlet Allocation (LDA) [17]. LDA is an unsupervised learning approach focusing on discovering hidden thematic structures in large text corpora. One of the issues associated with existing topic models is its inability to label the topics, limiting their usage in end-user applications such as visualisations. It is challenging to label discussion topics due to a lack of a reference source. As a solution, we proposed an automated topic labeling approach by generating candidate topic labels from course lectures. A Naïve Bayes classifier was trained to classify discussion topic into a week or set of weeks, and document summarisation techniques were applied to obtain the most suitable labels for each topic cluster. Our approach facilitates classifying and labeling the discussion threads using course lectures.

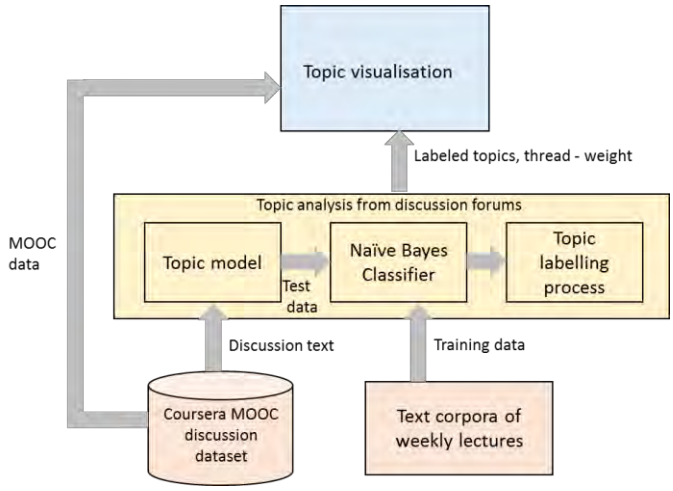


Figure 1: Overview of topic analysis and visualisation

We conducted experiments to evaluate our topic analysis approach using Machine Learning (ML), Statistics (STAT) and Psychology (PSY) MOOCs offered during 2013. In each course, we analysed approximately 5448, 2530 and 9384 number of posts and obtained 40, 25 and 40 strong topics for human annotation, respectively. Three human experts from each MOOC were recruited to label the topics manually and their mean inter-rater agreement (Kappa) was obtained as 0.75 (SD=0.09), 0.77 (SD=0.07) and 0.69 (SD=0.07) for ML, STAT and PSY respectively. We calculated the effectiveness of automated topic labeling process and obtained F-measure of 0.702, 0.75 and 0.69 for ML, STAT and PSY, respectively, demonstrating that the human-machine agreement is similar or slightly lower than inter-rater agreement. Our classifiers also performed well with a macroaveraged F-measure of 0.946, 0.926 and 0.896 for ML, STAT and PSY courses respectively We also calculated Mean Average Precision (MAP) to evaluate the ranked retrieval results of machine and obtained 0.806 (ML), 0.869 (STAT) and 0.774 (PSY). The promising results obtained from three MOOCs demonstrate that the proposed approach is effective for topic analysis of discussion contents.

3.2 Topic Visualisation

The design of our open source topic visualisation dashboard is motivated by the visual analytics process defined by Keim et al. [3] as "Analyze first, Show the Important, Zoom, filter and analyze further, Details on demand". Accordingly, our design includes analysing discussion topics, showing an overview of topic visualisation, filtering using different variables, analysing further using different variables, and providing details of individual threads on demand.

After identifying emergent topics from MOOC discussion contents (see Section 3.1), the focus of the topic visualisation is to demonstrate the discussion topics in a meaningful way for the end users, in our case the course staff, to make useful pedagogical decisions.

Our main focus is to visualise emergent topics of each course and their relationship with discussion threads. A sample screen of our visualisation dashboard using Psychology course is shown in Figure 2. As shown in Figure 2, the dashboard consists of three components; graph area, configurations, and the source.

The topic analysis is visualised by a bubble 'graph' using a force-directed layout, with larger nodes as topics and smaller nodes residing inside topics as threads. Initially, topic nodes are color-coded and adjusted in size to support visual perception of the amount of threads being discussed by the given topic (i.e. topic-thread weight). Topics are labeled using corresponding course lectures (see Section 3.1), while the similar-sized threads are initially labeled using the amount of posts associated with them. Color sliders at the bottom of the graph indicate the variations of topic-thread weight.

The 'configuration' panel (top panel of right hand side) allows the users to customise the visualisation according to their desire. Data can be imported as a CSV file for visualisation. Primarily, the data file should contain topic labels, associated thread ids, topic-thread weight and the number of posts each thread contains. However, depending on the requirement of the user, they can explore additional data such as views, votes to explore more interesting patterns. Initially, we present 10 emergent topics, supporting the visual analytics approach by Keim et al. [3] which recommends showing an overview first. The end users are allowed to adjust the number of topics up to 39, allowing a large amount of topics to be visualised for the analysis. The rationale behind limiting the number of topics to 39 is to fit into the screen resolution and similarly, if the topic-thread weight is reasonably low, it is likely that weaker topics (i.e. topic-thread weight below 0.5) are not effectively being labeled using course lectures [12]. The configuration panel also supports an optional color picker. However, the system supports variation of blue color as default.

An interesting aspect of this visualisation is that the user can explore different visualisations by changing the variables such as votes, views, instructor replies, time, number of words in threads etc. The application of the filtering parameters will change the color of topic nodes and labels of thread nodes (e.g. number of views). However, the size of the topic node remains unchanged to represent the amount of discussions associated with the given topic. For instance, number of views are vary from blue (highest number of views) to white (low number of views) (see Figure 4).

The 'source' panel provides detailed information of each thread on demand without overloading the visualisation. Users are allowed to click each 'thread' to select it and the discussions associated with this thread is shown in the bottom of the right hand side panel. In these visualisations, we have removed any identifiable data such as user or thread information.

Our open source dashboard is currently supported as a web-based system as well as standalone system which we intend to extend as a plugin embedded to the MOOC platforms.

We encounter repeated topic labels when more topic clusters are being labeled as corresponding to the same lecture. However, it is possible that these repeated topics are being discussed in slightly different threads depends on the distribution of topic terms within the topic model. If more than one topic ends up having the same label, we adjust the size of that particular topic to emphasise its' more strong influence as an emergent topic. It is also likely that a thread can be shared among multiple topics.

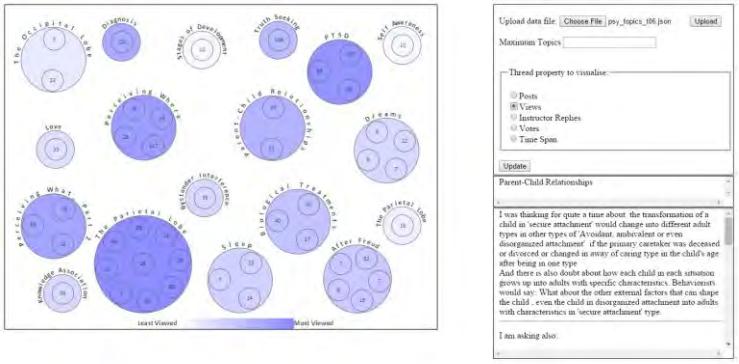


Figure 2: Topic Visualisation dashboard

It is important to determine the goals that are planned to be achieved using the visualisation in terms of improving teaching and learning within the educational context. With this in mind, we attempt to gain an understanding of online discourse at a massive scale by exploring the range of variables present in our interactive visualisation. The next section discusses our results along with interesting visualisations.

4. RESULTS AND DISCUSSION

4.1 Data

Our dataset include discussion contents (lecture-related) obtained from three MOOCs – *Machine Learning, Statistics: Making Sense of Data,* and *Psychology* within the Coursera platform with any user identification data removed (Table 1) [18].

Table 1. Statistics of selected MOOCs; ML-Machine learning, STAT-Statistics, PSY-Psychology

Course	Users *	Threads	Lecture-related threads	Total posts	Total words in threads	Mean (SD)
ML	6368	5449	972	5448	359,702	370 (229.6)
STAT	2313	1145	392	2530	155,329	396 (462)
PSY	11989	9300	1300	9384	719,797	553 (1014.6)

* Anonymous users are counted as 1 unit, so the number of actual discussion participants may be larger

4.2 Results

To identify emergent discussion topics of each MOOC, and as described in our earlier work, we applied Latent Dirichlet Allocation [17] and obtained 'unlabeled' topic clusters represented by a set of topic terms (usually using 10 terms). Further, we obtained list of threads associated with a given topic and their topic-thread weight (i.e. the proportion of the thread that contains the topic). From this, topics whose topic-thread weight less than 0.5 were eliminated due to the production of weak topics which mostly contain domain independent terms [12]. We apply our topic labeling mechanism to the filtered data in order to cluster the discussion topics using corresponding course lectures (see Section 3.1). Figure 3 demonstrates a sample screenshot (graph area only) obtained from our dashboard to answer the first research question in identifying emergent discussion topics.

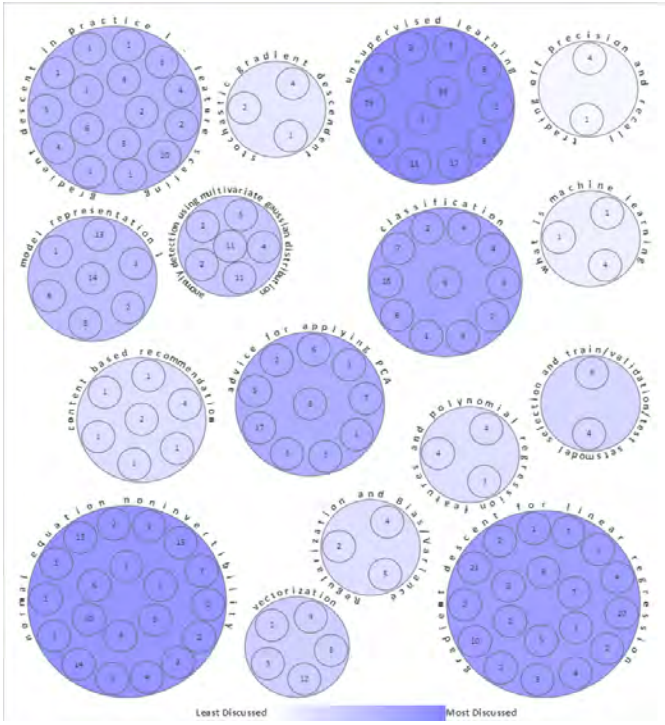


Figure 3: Sample topic-thread visualisation of Machine Learning course

As shown in Figure 3, we sized topic nodes (i.e. radius/diameter) in proportion to the number of threads to which the topic is associated, and the color in proportion to amount of posts. Even though the two topics 'content based recommendation' and 'model representation 1' are similar in size (i.e. 7 threads are associated with them), they vary in color. This occurs when the amount of posts is higher (i.e. 47) in the 'model representation 1' topic, emphasising that this topic is more thoroughly discussed by a relatively larger number of posts. The visualisation of topic-thread relationship facilitates in identifying the emergent topics as well as the topics that need teacher interventions. This visualisation of 'topic-wise classification' also assists experts in different 'topic' areas (e.g. community TAs or skilled participants) to jump into corresponding discussions and respond or assist the learners (see Figure 2 for source of thread texts). The visualisation of least discussed topics (depicted in 'lower resolution blue/white' color) assist in identification of the problematic topics for individual users or small set of users. Our approach in classifying the topics based on course lectures will also help teachers to ignore or deprioritise discussions that do not relate to course contents (e.g. social matters).

We explore topic-thread visualisation further by manipulating different variables relevant to our data including views, votes, and instructor replies to identify interesting patterns and correlations. This analysis answers the following questions;

1. What is the relationship between topics and votes?

The results of our statistical analysis show that the discussion topics and votes have a moderate positive correlation (r = 0.33; $p>0.01$) in Machine Learning course while no or negligible relationship (r = 0.13; $p>0.01$) in Statistics course since some participants tend to 'down vote' some discussions. However, Psychology course demonstrates a very strong positive correlation (r = 0.70; $p<0.01$). Thus, within the context of ML and STAT, the most discussed topics are not the ones most voted. For instance, Figure 4 demonstrates that the topic 'centre of the data and the effects of extreme values' is the most discussed topic, however, obtained only 1 vote, whereas 'data collection – observational studies' is one of those least discussed topics which obtained 8 votes. A higher number of votes suggests that the participants have more interest towards the topic or they are expecting much attention from the instructors, however, they may have less confidence to discuss it, perhaps due to lack of knowledge. Figure 4 visualises these findings.

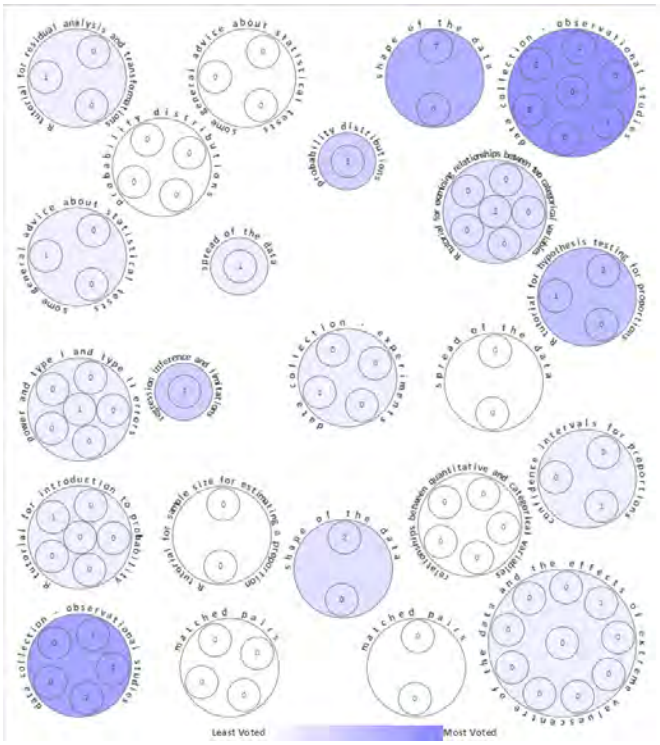


Figure 4: Relationship between topics and votes in the Statistics course

2. What is the relationship between topics and views?

We measured the correlation between the discussion topics and amount of views using Pearson correlation coefficient (r) and obtained statistically significant correlation; r = 0.7065 ($p<0.01$) for ML, r = 0.699 ($p<0.01$) for STAT and r = 0.79 ($p<0.01$) for PSY. This results suggest that the participants demonstrate more

interest towards emergent topics by viewing them more often. Similarly, less popular topics are viewed infrequently. Figure 5 depicts the visualisation correspond to this statistical analysis using the Machine Learning course.

According to the Figure 5, most discussed topics are illustrated by the size of the topic node while the most viewed topics are depicted using 'higher resolution blue' as shown in the color slider. The thread nodes are labeled using the number of views. Therefore, it is observable that the mostly discussed topics are similar to the mostly viewed topics in the Machine Learning course and vice versa. For instance, 'gradient descent for linear regression' and 'normal equation noninvertibility' are mostly discussed topics (determined by the size of the topic node) and they are also viewed more than thousand times. This kind of visualisation in classifying discussions according to topics will prioritise which posts to view and interact with based on specific requirements, resulting in a significant saving of time for both learners and teachers, particularly when reviewing massive amounts of data.

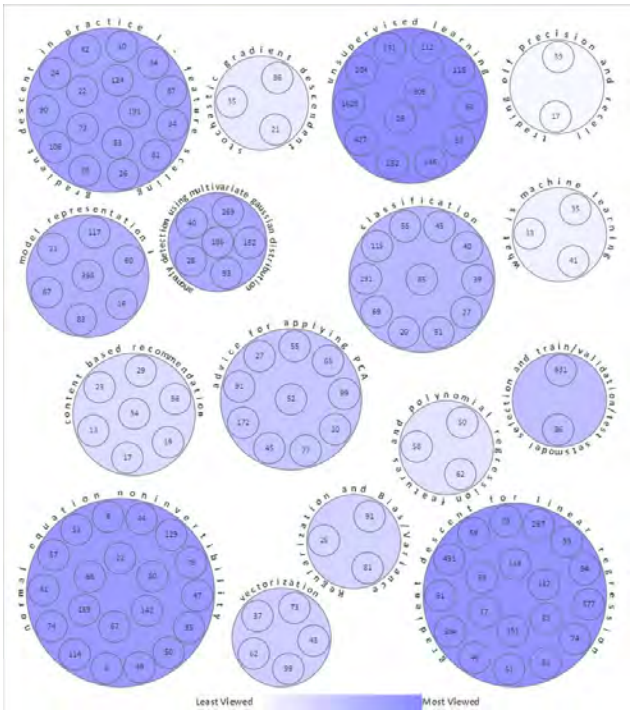


Figure 5: Relationship between topics and views in the Machine Learning course

3. What is the relationship between topics and instructor replies?

Instructor replies and discussion topics are moderately positively correlated in ML (r = 0.32; $p>0.01$). However, in STAT and PSY, these two variables demonstrate statistically significant results (r = 0.72; $p<0.01$ for STAT and r = 0.77; $p<0.01$ for PSY). This suggests that the instructors' intervention is more towards emergent topics which may isolate participants who have posted in other topics (i.e. declining topics). A study conducted by Dawson found that instructors primarily interact with high performing students despite isolated and low performing students being neglected irrespective of what they posts [8]. The ML course had relatively low instructor involvement for any topics while STAT and PSY courses had a good turnaround and strong positive correlation between these two variables. The visualisation in the Figure 6 demonstrates which topics require more inputs from instructors.

This analysis supports the open question of whether the emergent topics or declining topics require more instructor intervention. However, topic-wise classification will provide benefits to the instructors in identifying and prioritise the intervention. Simultaneously, a mechanism to 'pin' the emergent discussions will aid to avoid repeated discussions on the same topic.

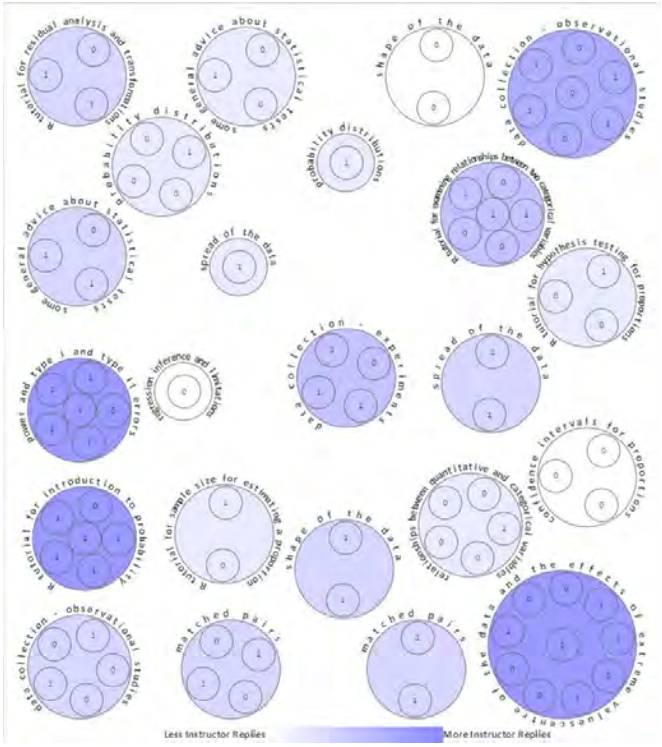


Figure 6: Relationship between topics and instructor replies in the Statistics course.

Our visualisation is currently extending to demonstrate the evolution of topics over time. The time-series analysis focuses on identifying corresponding week or set of weeks a given topic is being discussed. Some topics are discussed outside the course span (e.g. 'diagnosis' of Psychology course is discussed in week 9 where the course spans over 8 weeks). Timeline visualisation is helpful in identifying the topics that are being discussed either within or outside of the allocated weeks, enabling the identification of topics that are sustained throughout the course span.

This paper includes only a sample of visualisations and we have shared more visualisations based on the identified dataset here[3].

In summary, topic-thread visualisation assists in understanding massive volumes of discussion data by identifying emergent discussion themes, allowing the forum facilitators to make interventions more quickly rather than by reading and responding to individual threads. Similarly, topic-wise classification is supportive of comparison across discussions in understanding unexpectedly popular topics even after their expected periods in discussion.

The work presented in this paper is intended for MOOC course staff. We believe it will reduce manual forum moderation time in answering repeated questions, allowing novel discussions to occur contributing to new knowledge construction. Despite providing valuable insights into the analysis of large scale discourse, there is still considerable room for future research. These kinds of visualisation may also provide benefit to students, depending on their experience in interpreting visual information. Therefore, we consider that a topic-wise classification of discussion posts is useful as a navigational support for students, and intend to extend this work in future to support personalised navigation and recommendation of relevant posts.

This work does not yet include an in-depth analysis of individual topics or relationship between topics. It is yet to be analysed for relationship between topics and users. Our future work will include social network analysis to identify topic-inspired interactions between learner-teacher and learner-leaner (i.e. peers).

5. CONCLUSION

One of the primary challenges of MOOCs is to understand the massive volume of data to make inferences regarding student engagement or learning. To support this, our work analyses learner-generated discussion contents to identify emergent topics of discussions and labels them corresponding to the course lectures. This paper presents the visualisation of our topic-wise classification of discussion data, allowing the user to explore the analysis by manipulating different variables such as votes, views, instructor replies, and time-series analysis. A series of statistical analysis were performed to measure the correlation between discussion topics and other variables, and the finding were compared using the visualisation dashboard. This work provides benefit to the educational data mining and learning analytics research community through an open framework for topic analysis and visualisation of massive volume of discussion data generated regularly through MOOCs and other online learning platforms.

6. ACKNOWLEDGMENTS

The authors would like to acknowledge Google Inc. for supporting this project through the 'Google MOOC Focused Research Award Scheme'.

7. REFERENCES

[1] Thomas, J.J. and K.A. Cook, *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. 2005: IEEE Computer Society Press.

[2] Coffrin, C., et al., *Visualizing patterns of student engagement and performance in MOOCs*, in *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge*. 2014.

[3] Keim, D., et al., *Visual Analytics: Definition, Process, and Challenges*, in *Information Visualization*. 2008, Springer Berlin Heidelberg. p. 154-175.

[4] Arnold, K.E. and M.D. Pistilli, *Course signals at Purdue: using learning analytics to increase student success*, in *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. 2012.

[5] Mercer, N., *Sociocultural discourse analysis: analysing classroom talk as a social mode of thinking.* Journal of Applied Linguistic, 2004. **1**(2): p. 137-168.

[6] Ezen-Can, A., et al., *Unsupervised modeling for understanding MOOC discussion forums: a learning analytics approach*, in *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*. 2015.

[7] Reich, J., et al., *Computer Assisted Reading and Discovery for Student Generated Text in Massive Open Online Courses.* Journal of Learning Analytics, 2015. **2**(1).

[8] Bakharia, A. and S. Dawson. *SNAPP: A Bird's-Eye View of Temporal Participant Interaction*. in *Proceeding of the 1st International Conference on Learning Analytics and Knowledge*. 2011.

[9] Oshima, J., R. Oshima, and Y. Matsuzawa, *Knowledge building discourse explorer: a social network analysis application for knowledge building discourse.* Educational technology research and development, 2012. **60**(5): p. 903-921.

[10] Chen, B., *Visualizing semantic space of online discourse: the knowledge forum case*, in *Proceedings of the Fourth International Conference on Learning Analytics And Knowledge*. 2014.

[11] Speck, J., et al., *ForumDash: analyzing online discussion forums*, in *Proceedings of the first ACM conference on Learning @ scale conference*. 2014.

[12] Atapattu, T. and K. Falkner. *A Framework for Topic Generation and Labeling from MOOC Discussions*. in *Third Annual ACM conference on Learning at Scale*. 2016.

[13] Xu, Z., et al., *Visual analytics of MOOCs at maryland*, in *Proceedings of the first ACM conference on Learning @ scale conference*. 2014.

[14] Teplovs, C., N. Fujita, and R. Vatrapu, *Generating predictive models of learner community dynamics*, in *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. 2011.

[15] Liddo, A., et al., *Discourse-centric learning analytics*, in *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. 2011.

[16] Hsiao, I. and P. Awasthi, *Topic facet modeling: semantic visual analytics for online discussion forums*, in *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*. 2015.

[17] Blei, D.M., A.Y. Ng, and M.I. Jordan, *Latent Dirichlet Allocation.* Journal of Machine Learning Research, 2003. **3**: p. 993-1022.

[18] Rossi, L.A. and O. Gnawali. *Language Independent Analysis and Classification of Discussion Threads in Coursera MOOC Forums*. in *Proceedings of the IEEE International Conference on Information Reuse and Integration (IRI 2014)*. 2014.

Document Segmentation for Labeling with Academic Learning Objectives

Divyanshu Bhartiya
IBM Research
Bangalore, India
dibharti@in.ibm.com

Danish Contractor
IBM Research
New Delhi, India
dcontrac@in.ibm.com

Sovan Biswas
IBM Research
Bangalore, India
sobiswa3@in.ibm.com

Bikram Sengupta
IBM Research
Bangalore, India
bsengupt@in.ibm.com

Mukesh Mohania
IBM Research
Melbourne, Australia
mukeshm@au1.ibm.com

ABSTRACT

Teaching in formal academic environments typically follows a curriculum that specifies learning objectives that need to be met at each phase of a student's academic progression. In this paper, we address the novel task of identifying document *segments* in educational material that are relevant for different learning objectives. Using a dynamic programming algorithm based on a vector space representation of sentences in a document, we automatically segment and then label document segments with learning objectives. We demonstrate the effectiveness of our approach on a real-world education data set. We further demonstrate how our system is useful for related tasks of document passage retrieval and QA using a large publicly available dataset. To the best of our knowledge we are the first to attempt the task of segmenting and labeling education materials with academic learning objectives.

Keywords

text segmentation, document labeling, academic learning objectives, unsupervised

1. INTRODUCTION

The rapid growth of cost-effective smart-phones and media devices, coupled with technologies like Learning Content Management Systems, tutoring systems, digital classrooms, MOOC based eLearning systems etc. are changing the way today's students are educated. A recent survey [1] found that there was a 45% year-on-year uptake between 2013 and 2014 of digital content in the classroom and a nearly 82% uptake in the use of digital textbooks. Of the 400,000 K-12 students surveyed, 37% of them reported using online textbooks for their learning needs. Students and teachers frequently search for free and open education resources available online to augment or replace existing learning material. Organizations like MERLOT[2] and the Open Education Consortium[3] offer and promote the use of free learning resources by indexing material available on the web, based only on keywords or user specified meta-data. This makes the identification of the most relevant resources difficult and time consuming. In addition, the use of manually specified meta-data can also result in poor results due to inconsistent meta-data quality, consistency and coverage. Identifying materials most suitable for a learner can be aided by tagging them with learning objectives from different curricula. However, manually labeling material with learning objectives is not scalable since learning standards can contain tens of hundreds of objectives and are prone to frequent revision. Recent work by [3] attempted to address this problem by using external resources such as Wikipedia to expand the context of learning objectives and a *tf-idf* based vector representation of documents and learning objectives. One of the limitations of the system is that it works well only when documents are relatively short in length and relate to a few learning standard objectives. The accuracy of the algorithm reduces when the documents considered are resources such as textbooks due to the dilution of the weights in the *tf-idf* based vector space model. Further, from the perspective of information access, returning a large reference book for a learning objective still burdens the user with the task of identifying the relevant portions of the book. This, therefore, does not adequately address the problem.

In this paper, we address the problem of finding document segments most relevant to learning objectives, using document segmentation [1] and segment ranking. To the best of our knowledge, we are the first to attempt the problem of segmenting and labeling education materials with academic learning objectives.

In summary, our paper makes the following contributions:

- We define the novel task of identifying and labeling document segments with academic learning objectives.

[1]Project Tomorrow, Trends in Digital Learning 2015

[2]http://www.merlot.org
[3]http://www.oeconsortium.org/

- We present the first system that identifies portions of text most relevant for a learning objective in large educational materials. We demonstrate the effectiveness of our approach on a real world education data set. We report a sentence level $F1$ score of 0.6 and a segment level minimal match accuracy@3 of 0.9
- We demonstrate, using a large publicly available dataset, how our methods can also be used for other NLP tasks such as document passage retrieval and QA.

The rest of the paper is organized as follows: In the next section we describe related work, in section 3 we formally describe our problem statement, section 4 describes our algorithm and implementation details and section 5 presents our detailed experiments. Finally, in section 6 we conclude this paper and discuss possible directions of future work.

2. RELATED WORK

Broadly, our work is related to three major areas of natural language research: Text Segmentation, Query Focused Summarization and Document Passage Retrieval. We present a comparison and discussion for each of these areas below:

Text Segmentation: Typically, the problem of automatically chunking text into smaller meaningful units has been addressed by studying changes in vocabulary patterns [6] and building topic models[5]. In [12], the authors adapt the TextTiling algorithm from [6] to use topics instead of words. Most recently, [1] uses semantic word embeddings for the text segmentation task. While supervised approaches tend to perform better, we decided to adapt the state of the art unsupervised text segmentation method proposed in [1], due to the challenges associated with sourcing training data for supervised learning.

Query Focused Summarization: Focused summarization in our context [8], [10] [4] is the task of building summaries of learning materials based on learning objectives. Here, each learning objective can be treated as a *query*, and the learning materials as documents that need to be summarized. However, it is important to note that in the education domain, any such summarization needs to ensure that summarized material is presented in a way that facilitates learning. This poses additional research challenges such as automatically identifying relationships between concepts presented in the material and therefore, in this paper, we do not model our problem as a summarization task. We encourage the reader to consider it as a possible direction for future research.

Document Passage Retrieval: Lastly, document passage retrieval [2] is the task of fetching relevant document passages from a collection of documents based on a user query. However, such tasks typically require the passage boundaries to be well known and therefore, cannot return sub-portions that may be present within a passage or return results that span sub-parts of multiple passages.

3. PROBLEM STATEMENT

Typically, a learning standard consists of a hierarchical organization of learning objectives where learning objectives are grouped by Topic, Course, Subject and Grade. For the purpose of this paper we refer to a "label" as the complete Grade (g) -> Subject (s) -> Course (c) -> Topic (t) -> Learning Objective (l) path in the learning standard.

Given a document $\mathcal{D}$ of length N we would like to identify the most relevant segments $\phi_{ij}^{\{g,s,c,t,l\}}$ for a given label $\{g, s, c, t, l\}$ where i, j denote positions in a document i.e $i, j \in [0, N]$ and $i < j$. In the rest of the paper, we denote the learning objective $\{g, s, c, t, l\}$ as e to ease notation.

Figure 1 shows chapter 2 from the the "College Physics" OpenStax textbook[4]. The segments (demarcated using rectangles) have been identified for two learning objectives INST1 and INST2 and occur in different portions of the book. They can even be a sub-part of an existing section in a chapter as shown for INST1.

The next section describes our algorithm for the problem of segmentation and labeling based on learning objectives.

4. OUR METHOD

We represent each sentence as a unit vector s_i, $(0 \leq i \leq N-1)$ in a Dim dimensional space. The goal of segmentation is to find K splits in a document, denoted by $(x_0, x_1, \ldots, x_K)$, where $x_0 = 0$ and $x_K = N$ and x_i denotes the line number specifying the segment boundary such that if the kth segment contains the sentence s_i, then $x_{k-1} \leq i < x_k$. The discovered segment $\phi_{i,j}$ is the segment between the splits x_i and x_j . Depending on the granularity of the learning objectives and the document collection, the optimal number of splits can be set (See section 5). Let the cost function ψ for a segment $\psi(i, j)$ measure the *internal cohesion* of the segment, $(0 \leq i < j \leq N)$. The segmentation score for K splits $s = (x_0, x_1, \ldots, x_K)$ can then be defined as Ψ :

$$\Psi(s) = \psi(x_0, x_1) + \psi(x_1, x_2) + \ldots + \psi(x_{K-1}, x_K)$$

To find the optimal splits in the document based on the cost function Ψ, we use dynamic programming. The cost of splitting $\Psi(N, K)$ is the cost of splitting 0 to N sentences using K splits. So,

$$\Psi(N, 1) = \psi(0, N)$$

$$\Psi(N, K) = \min_{l<N} \Psi(l, K-1) + \psi(l, N)$$

We define the ψ function as follows:

$$\psi(i, j) = \sum_{i \leq l < j} \|s_l - \mu(i, j)\|^2$$

where $\psi(i, j)$ is analogous to the intra-cluster distance in traditional document clustering while $\mu(i, j)$ is a representative vector of the segment. We discuss possible forms of μ later in this section.

Ranking: Each segment is represented as a normalized vector $\mu(i, j)$ and we determine the most relevant segments to a learning objective e by ranking segments in increasing order of similarity based on cosine similarity.

$$\cos(\mu, e) = \sum_{d=1}^{Dim} \mu_d * e_d$$

[4]https://openstax.org/details/college-physics

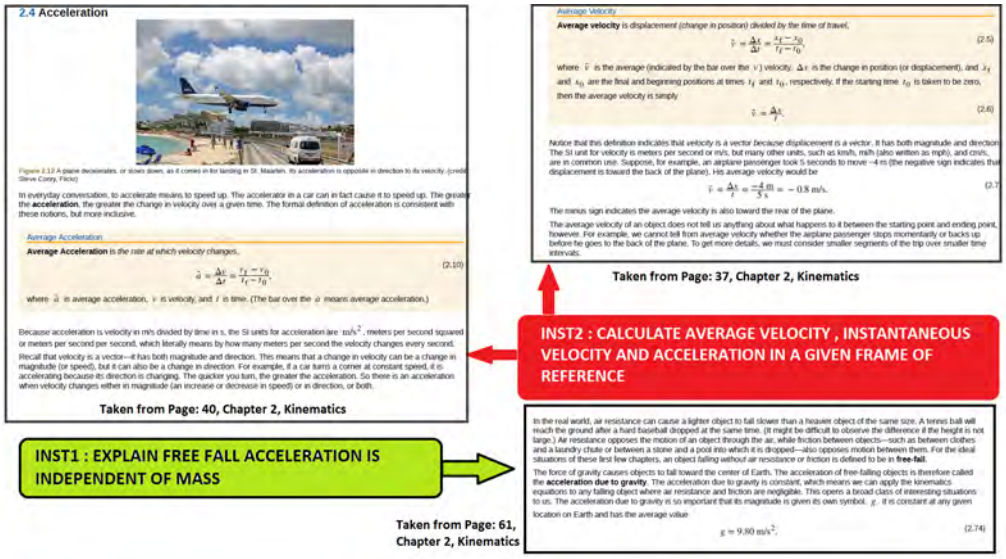


Figure 1: This image shows excerpts from chapter 2 Kinematics from the College Physics text book by OpenStax along with the segment boundaries for two learning objectives INST1 and INST2 shown in colors red and green respectively.

We then select the top n ranked segments as the segments relevant to the learning objective. In section 5.3 we describe how the number of splits K as well as the value of n can be chosen empirically given a validation data set.

We now describe different methods of constructing the document and segment vectors:

TF-IDF: Each sentence is represented as a bag of words, the dimensionality being the vocabulary size. Each word in a sentence v_i is weighted by its *tfidf* measure. For a word v_i in the sentence s_k of a document $\mathcal{D}$, the *tfidf* measure is given by :

$$tfidf(v_i)_{s_k,\mathcal{D}} = f(v_i, \mathcal{D}) \log\left(\frac{|D|}{df(v_i)}\right)$$

where $f(v_i, d)$ is the frequency of the word v_i in the document d, $|D|$ being the total number of documents in our corpus and $df(v_i)$ is the number of documents with the word v_i in it. The segment vector $\mu(i, j)$ in this case is the mean of the sentence vectors in that segment.

Word Vector: We represented each sentence as a weighted combination of the word vectors in a sentence. The word-vector w_i for each word v_i is specified using Mikolov's Word2Vec[9]. Each sentence s_i is represented as:

$$s_i = \sum_v f(v, d) \log\left(\frac{|D|}{df(v)}\right) w_i$$

The segment vector $\mu(i, j)$ is also the mean vector in this case.

Fisher Vector: Paragraph vectors[7] try to embed the sentences in a fixed dimension, but they require extensive training on the source dataset. Instead we use Fisher Vectors, which have been widely used in the vision community [11] for combining different feature vectors (word vectors in our case), and were recently used for question retrieval by Zhou et.al. [15]. The word vocabulary is modeled as a Gaussian Mixture Model, since a GMM can approximate any continuous arbitrary probability density function. Let $\lambda = \{\theta_j, \mu_j, \Sigma_j, j = 1 \dots N_G\}$ be the parameters of the GMM with N_G gaussians. Let, $\{w_1, w_2, \dots, w_T\}$ be the vectors for the words $v_1, v_2, \dots, v_T$ in the sentence s_i for which we need to construct the fisher vector. We define $\gamma_j(w_t)$ to be the probability that the word w_t is assigned the gaussian j,

$$\gamma_j(w_t) = \frac{\theta_j \mathcal{N}(w_t|\mu_j, \Sigma_j)}{\sum_{u=1}^{N_G} \theta_u \mathcal{N}(w_t|\mu_u, \Sigma_u)}$$

We define the gradient vector as the score for a sentence, $G_\lambda(s_i)$ [13]. To compare two sentences, Fisher Kernel is applied on these gradients,

$$\mathcal{K}(s_i, s_j) = G_\lambda(s_i) F_\lambda^{-1} G_\lambda(s_j)$$

where, F_λ is the Fisher Information Matrix,

$$F_\lambda = E_{x \sim p(x|\lambda)}[G_\lambda(s_i) G_\lambda(s_j)^T]$$

F_λ^-1 can be decomposed as $L_\lambda^T L_\lambda$, hence the Fisher Kernel can be decomposed to two normalized vectors, $\Gamma_\lambda(s_i) = L_\lambda G_\lambda(s_i)$. This $\Gamma_\lambda(s_i)$ is the fisher vector for the sentence s_i

$$\Gamma_{\mu_j^d}(s_i) = \frac{1}{T\sqrt{\theta_j}} \sum_{t=1}^{T} \gamma_j(w_t) \left(\frac{w_t^d - \mu_j^d}{\sigma_j^d}\right) \quad (1)$$

$$\Gamma_{\sigma_j^d}(s_i) = \frac{1}{T\sqrt{2\theta_j}} \sum_{t=1}^{T} \gamma_j(w_t) \left[\frac{(w_t^d - \mu_j^d)^2}{(\sigma_j^d)^2} - 1\right] \quad (2)$$

The final fisher vector is the concatenation of all $\Gamma_{\mu_j^d}(s_i)$ and $\Gamma_{\sigma_j^d}(s_i)$ for all $j = 1 \dots N_G$, $d = 1 \dots Dim$, hence $2 * N_G * Dim$ dimensional vector. We define the segment vector $\mu(i, j)$ as the fisher vector formed by using the word vectors

in the segment, hence giving us a unified representation of the segment.

5. EXPERIMENTS

In this section we evaluate our method for identifying document segments suited for learning objectives.

5.1 Data

We made use of two data sets for our experiments:

AKS labeled Data Set: We use the collection of 110 Science documents used by [3] labeled with 68 learning objectives from the Academic Knowledge and Skills (AKS)[5]. We also used term expansions as described in [3] to increase the context of learning objectives. We further identified document segments (at the sentence level) suitable for the learning standard in each of the documents, where applicable.

To build a collection of documents covering multiple learning objectives, we simulated the creation of large academic documents such as text books, by augmenting each lecture note with 9 randomly selected lecture notes. Thus, for each of the 68 instructions that were covered in our data set, we created 5 larger documents each consisting of 10 documents from the original set, giving us a document collection of 340 large documents, with an average length of 300 sentences.

Dataset	#Docs	#Avg. Sentences	#Avg. Splits
AKS Dataset	340	300	10
WikiQA	8100	180	10

WikiQA Dataset: To show the general applicability of our approach on tasks such as document passage retrieval and QA, we also use the recently released WikiQA data set [14] which consists of 3047 questions sampled from Bing[6] query logs and associated with answers in a Wikipedia summary paragraph. As outlined in the approach above, for each of the questions, we created a larger document by including 9 other randomly selected answer passages. For each of the 2700 questions from the Train and Test collection we created 3 such documents, thus giving us 8100 documents.

5.2 Evaluation Metrics

We define the following metrics for our evaluation:

MRR (Mean Reciprocal Rank) : The MRR is defined as the reciprocal rank of the of the first correct result in a ranked list of candidate results.

P@N (Precision@N): Let the set of sentences in the top N segments identified be Γ^{Sys} and further, let the set of sentences in the gold standard be Γ^{Gold}. The precision@N is given by :

$$P@N = \frac{|\Gamma^{Sys} \cap \Gamma^{Gold}|}{|\Gamma^{Sys}|} \tag{3}$$

R@N (Recall@N):Using the same notation described above, the recall @ N is given by :

$$R@N = \frac{|\Gamma^{Sys} \cap \Gamma^{Gold}|}{|\Gamma^{Gold}|} \tag{4}$$

F1@N (F1 Score @N): The F1 Score@N is given by the harmonic mean of the Precision@N and Recall@N described above. **MMA@N (Minimal Match Accuracy@N)** For a collection of D labeled documents, the minimal match accuracy@N is a relaxed value of precision and is given by:

$$\frac{\sum_i^D \mathbb{1}\{|\Gamma_i^{Sys} \cap \Gamma_i^{Gold}| \geq 1\}}{D} \tag{5}$$

where $\mathbb{1}\{\}$ is the indicator function.

5.3 Experimental Setup

For the AKS dataset, we calculate the *idf* using a collection of 6000 Science documents from Wikibooks[7] and Project Gutenburg[8]. For the WikiQA dataset, *idf* was calculated on the 2700 summaries in the training and test collection. Word vectors and fisher vectors were trained on the full collection of English Wikipedia articles to ensure that the Gaussian Mixture model isn't trained on a skewed dataset and can be used across universally for all kinds of english educational documents. The number of gaussians were selected based on the bayesian information criterion.[9]

Choosing the number of top segments: The number of top ranked segments n and the number of splits K both affect the accuracy of the system. For instance, if we set K to be half the total number of sentences, the resulting segments will be very small. Therefore, the value of n needs to be higher to have adequate coverage (recall). Similarly, choosing very few splits can result in large chunks, which can be problematic if the learning objectives were precise and required finer segments. Thus, the choice of n and K depends on the granularity of specification in the learning objectives as well as the nature of content in the document collection.

We use 20% of the dataset (selected at random) as the validation set for tuning the parameters n and k. By varying both n and K we can determine the value at which the system performance (measured using F1 score) is best. Figure 2 shows the variation in F1 Score for different values of K and n. For clarity of presentation, we only show this for the system using TF-IDF vectors. As can be seen, the $F1$ score is best for 10 splits and choosing the 3 best segments closest to the learning objective i.e $K = 10, n = 3$. Figures 3 and 4 show the individual contributions to the $F1$ score.

5.4 Results

5.4.1 Document Segmentation and Labeling

On performing segmentation on the AKS dataset using all three vector approaches, we observe (table 1) that the tf-idf vector representation works best. We noticed that many

[5]https://publish.gwinnett.k12.ga.us/gcps/home/public/parents/content/general-info/aks
[6]http://www.bing.com
[7]http://www.wikibooks.org
[8]http://www.gutenburg.org
[9]An index used for model selection $-2L_m + m\ln n$, where L_m is the maximized likelihood, m are the number of parameters and n is the sample size

Query Expansion		@1			@3			@5		
		P	R	F1	P	R	F1	P	R	F1
No Expansion	TFIDF	0.669	0.359	0.468	0.493	0.698	**0.578**	0.395	0.843	0.538
	WORDVEC	0.462	0.357	0.403	0.331	0.633	0.434	0.284	0.829	0.423
	FISHER	0.476	0.366	0.414	0.342	0.679	0.454	0.284	0.855	0.426
With Expansion	TFIDF	0.686	0.320	0.436	0.545	0.701	**0.613**	0.435	0.856	0.577
	WORDVEC	0.483	0.323	0.387	0.351	0.586	0.439	0.308	0.797	0.444
	FISHER	0.481	0.322	0.386	0.351	0.619	0.448	0.305	0.827	0.445

Table 1: Results on the AKS Labeled Dataset

	MRR	MMA@1	MMA@3	MMA@5
TFIDF	0.78	0.652	**0.905**	0.882
WORDVEC	0.56	0.429	0.635	0.782
FISHER	0.55	0.405	0.620	0.715

Table 2: Segment Level Results on AKS Dataset

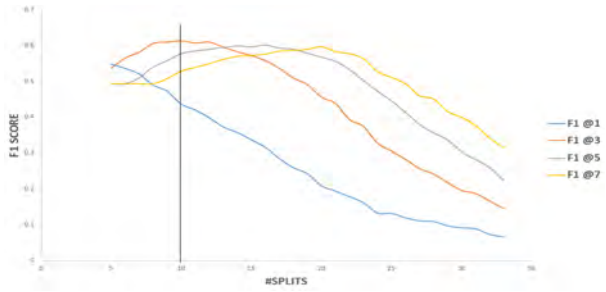


Figure 2: F1 Variation with number of segments at varying depths of retrieval. Best score at 10 segments at depth 3

of the documents in the AKS data set were very well contextualized when changing topics, thus blurring the segment boundaries. For example, in one of the documents which described "Motion in a Straight Line", the concepts of "velocity", "acceleration", "position-time" graphs are intertwined and the topical drift is not easy to observe. As a result, due to the nature of documents in the collection, we hypothesize that the fisher vectors and word vectors which have been trained on large general corpora are unable to adequately distinguish some portions of the text, while the tf-idf vectors which have been tuned on the corpus better reflect the word distributions.

The precision, recall and F1 scores are calculated at the sentence level, thus making it a very strict measure. So we also report segment level accuracy, i.e. how many of the top n segments identified were relevant. A predicted segment is labeled relevant to the external query if at least 70% of the segment overlaps with the gold labeled segments. We evaluate the performance using MRR and MMA@N. Table 2 shows the segment level evaluation of our system.

5.4.2 *Passage Retrieval and QA*

We also conducted experiments with a more discriminative dataset where the topical shift is not as hard to observe. We report (table 3) an MRR of 0.895 and P@1 of 89.4% for the passage retrieval task on each of the documents generated,

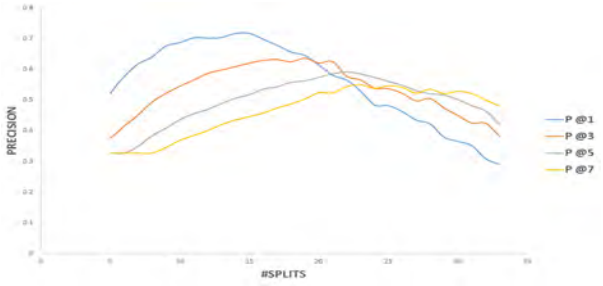


Figure 3: Precision variation with number of segments at varying depths of retrieval. Low values of n and high values of K give high precision. Increasing K while keeping n constant gives a drop in precision.

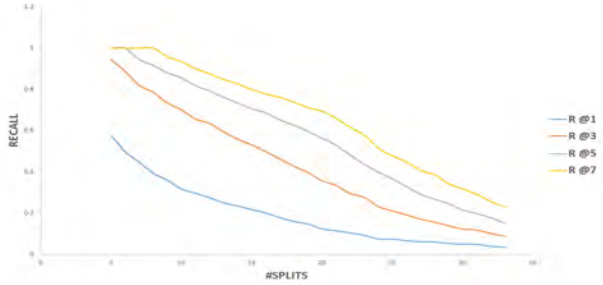


Figure 4: Recall variation with number of segments at varying depths of retrieval. Recall is higher at low values of K and high values of n, and the recall drops considerably as the number of segments K increases.

as described in section 5.1.

Further, we also describe our results on the original task, proposed with the data set, of finding the answer in a passage for a question. In our experiments we report results under two conditions: (a) First identifying the best passage and then choosing the best sentence (b) Assuming the best passage is already known and then choosing the best sentence that answers the query (original WikiQA QA task). Table 4 presents results of experiments under both these conditions. It can be seen that our system gives comparable results under both conditions. The state of the art results under condition (b) as reported in the original paper is an MRR of 0.696. Our system, though not designed for the original task, has an MRR score 10% lower than the best system reported.

				@1			@3		
	MRR	**MMA@1**	**MMA@3**	**P**	**R**	**F1**	**P**	**R**	**F1**
TFIDF	0.807	0.797	0.812	0.839	0.893	0.865	0.308	0.958	0.466
WORDVEC	**0.895**	0.877	0.913	**0.894**	0.914	0.904	0.315	0.984	0.478
FISHER	0.865	0.842	0.887	0.863	0.885	0.874	0.298	0.975	0.457

Table 3: WikiQA Passage Retrieval Results

	MRR Top Segment	**MRR Gold Standard Passage**
TFIDF	0.528	0.495
WORDVEC	0.548	0.586
FISHER	0.577	**0.597**

Table 4: Finding the sentence answering the question: "Top segment" uses our system to select the best passage while "Gold standard passage" uses the actual passage labeled in the data set

6. DISCUSSION AND CONCLUSION

In this paper we described the novel task of automatically segmenting and labeling documents with learning standard objectives. Using a state of the art dynamic programming algorithm for text segmentation, we demonstrate its use for this problem and report a sentence level $F1$ score of 0.613 and segment level $MMA@3$ of 0.9. We also demonstrated the effectiveness of our approach on document passage retrieval and QA tasks.

Our method is completely unsupervised and only requires a small validation set for parameter tuning. This makes our work general and easily applicable across different geographies and learning standards. Identifying document segments best suited for learning objectives is a challenging problem. For instance, portions of documents that introduce or summarize topics or build a background in an area are very hard to disambiguate for the algorithm due to the lack of observable topic shifts. Developing more sophisticated cohesion and topical diversity measures to address this problem could be an interesting direction of further research.

In future work, we would also like to explore methods that jointly segment and label documents. We also plan to use other methods of vector construction such as paragraph vectors [7] to better represent segments using a training data set as well as semi-supervised text segmentation methods.

7. REFERENCES

[1] A. A. Alemi and P. Ginsparg. Text segmentation based on semantic word embeddings. *arXiv preprint arXiv:1503.05543*, 2015.

[2] C. L. Clarke and E. L. Terra. Passage retrieval vs. document retrieval for factoid question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 427–428. ACM, 2003.

[3] D. Contractor, K. Popat, S. Ikbal, S. Negi, B. Sengupta, and M. K. Mohania. Labeling educational content with academic learning standards. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, pages 136–144, 2015.

[4] H. Daumé III and D. Marcu. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 305–312. Association for Computational Linguistics, 2006.

[5] L. Du, J. K. Pate, and M. Johnson. Topic segmentation in an ordering-based topic model. 2015.

[6] M. A. Hearst. Texttiling: A quantitative approach to discourse segmentation. Technical report, Citeseer, 1993.

[7] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[8] J.-P. Mei and L. Chen. Sumcr: a new subtopic-based extractive approach for text summarization. *Knowledge and information systems*, 31(3):527–545, 2012.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean. word2vec, 2014.

[10] Y. Ouyang, W. Li, S. Li, and Q. Lu. Applying regression models to query-focused multi-document summarization. *Information Processing & Management*, 47(2):227–237, 2011.

[11] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Computer Vision–ECCV 2010*, pages 143–156. Springer, 2010.

[12] M. Riedl and C. Biemann. Text segmentation with topic models. *Journal for Language Technology and Computational Linguistics*, 27(1):47–69, 2012.

[13] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.

[14] Y. Yang, W.-t. Yih, and C. Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018. Citeseer, 2015.

[15] G. Zhou, T. He, J. Zhao, and P. Hu. Learning continuous word embedding with metadata for question retrieval in community question answering. In *Proceedings of ACL*, pages 250–259, 2015.

Semi-Automatic Detection of Teacher Questions from Human-Transcripts of Audio in Live Classrooms

Nathaniel Blanchard[1], Patrick J. Donnelly[1], Andrew M. Olney[2], Borhan Samei[2], Brooke Ward[3], Xiaoyi Sun[3], Sean Kelly[4], Martin Nystrand[3], Sidney K. D'Mello[1]
[1]University of Notre Dame; [2]University of Memphis;
[3]University of Wisconsin-Madison; [4]University of Pittsburgh
384 Fitzpatrick Hall
Notre Dame, IN 46646, USA
nblancha@nd.edu; sdmello@nd.edu

ABSTRACT
We investigate automatic detection of teacher questions from automatically segmented human-transcripts of teacher audio recordings collected in live classrooms. Using a dataset of audio recordings from 11 teachers across 37 class sessions, we automatically segment teacher speech into individual teacher utterances and code each as containing a teacher question or not. We trained supervised machine learning models to detect questions using high-level natural language features extracted from human transcriptions of a random subset of 1,000 segmented utterances. The models were trained and validated independently of the teacher to ensure generalization to new teachers. We are able to detect questions with a weighted F_1 score of 0.66, suggesting the feasibility of question detection on automatically segmented audio from noisy classrooms. We discuss the possibility of using automatic speech recognition to replace the human transcripts with an eye towards providing automatic feedback to teachers.

Keywords
Automatic Speech Recognition, Natural Language Processing, Classroom Environments, Question Detection

1. INTRODUCTION
Teachers employ a wide array of instructional strategies in their classrooms due to individual teaching styles, requirements of the curricula, and other constraints. These strategies may include lectures, asking questions and evaluating student responses, or assigning small-group work, among many others. However, these approaches are not equally effective at promoting student achievement. Certain techniques, such as asking particular types of questions or facilitating a classroom-wide discussion, have been shown to predict student engagement and achievement growth above others [1], [2].

Research also indicates that providing teachers with feedback on their instructional practices can lead to improved student achievement [3]. But where does the feedback come from? Currently, the onus is on trained human judges who analyze teacher instruction by observing live classrooms. For example, the Nystrand and Gamoran coding scheme [4], [5] provides a general template for observers to document and analyze teacher instructional practices. This scheme has been empirically validated in numerous studies across hundreds of middle school and high school classrooms [6]–[8]. Unfortunately, this is an expensive and labor intensive process that hinders the ability to analyze classroom instruction at scale. Instead, computational methods that can automatically analyze classroom instruction at scale are needed. We take a step in this direction by considering the possibility of detecting teacher questions in live classrooms. We focus on questions because they are a central component of dialogic instruction, often serving as a catalyst for in-depth classroom discussions and so called 'dialogic spells' [9].

The classroom environment provides a unique set of challenges for the automatic analysis of questions. There are also numerous constraints as discussed in detail by D'Mello et al. [10]. Briefly, the analytic approach should not be disruptive to either the teacher or the students. Secondly, it must be affordable to enable wide-spread adoption across classrooms. Finally, for privacy concerns, video recordings are not possible unless students can be de-identified.

We attempted to overcome these challenges by designing a system that includes a low cost, wireless headset microphone to record teachers as they move about the classroom freely. Our system accommodates various seating arrangements, classroom sizes, and room layouts, but attempts to mitigate complications due to ambient classroom noise, muffled speech, or classroom interruptions, factors that reflect the reality of real-world environments.

There is the open question as to whether the data collected in this fashion can be of sufficient quality for automatic question detection. As an initial step, we consider semi-automated question detection from human-transcripts of automatically-segmented teacher audio. If successful, the next step would be to apply our basic approach by using automatic speech recognition (ASR) in lieu of human transcriptions.

1.1 Related Work
Our work is related to previous attempts at automatic detection of questions from transcriptions of audio albeit outside of the noisy classroom interaction context we consider here. We limit our review to experiments that include ASR, as our ultimate goal is in full automation of question detection.

In a study attempting to detect questions in office meetings, Boakye et al. [11] trained models using the ICSI Meeting Recorder Dialog Act (MRDA) corpus, a dataset of 75 hour-long meetings recorded with headset and lapel microphones. Using an AdaBoost classifier to detect questions from human transcriptions, the authors obtained an F_1 score of 67.6 by combining various NLP features.

Stolcke et al. [12] built a dialogic act tagger on the conversational switchboard database. A Bayesian network modeling word and trigrams discourse grammars, from human transcriptions achieved a recognition rate of 71% to detect a set of dialogic acts, such as statements, questions, apologies, or agreement (chance level 35%; human agreement 84%). The authors further attempted to distinguish questions from statements, two speech acts often confused by their model. They obtained an accuracy of 86% on a subset of their dataset containing equal proportions of questions and statements using only word features (chance accuracy 50%). This result, while promising, is based on an artificially balanced dataset of statements and questions.

Most recently, Orosanu and Jouvet [13] investigated classification of sentences labeled as either statements or questions in three French language corpora, testing on a set of 7,005 statements and 831 questions. The models accurately classified 75.5% of questions and 72.0% of statements using human transcripts. The authors compared the results of using human-annotated sentence boundaries against a semi-automatic method for boundary detection. A subset of sentences, those without prior and proceeding silences of an undefined length, were split once on the longest silence in the sentence; the remainder of the sentences were left unchanged. Semi-automatic splitting led to a 3% increase in classification errors. Although only a subset of sentences were split and there were no cases where sentences were combined, the results suggest that detecting questions from imperfect boundaries may be possible.

1.2 Contributions and Novelty

We describe an approach to automatically identify teacher questions from human-transcriptions of teacher audio recorded in live classrooms. We make several contributions while addressing these challenges. First, we examine a dataset of full length recordings of real world class sessions, drawn from multiple teachers and schools. Second, we only use teacher audio because it is the most scalable and practical option. Third, we automatically segment audio recordings into individual teacher utterances in a fully automated fashion and manually transcribe a subset of these utterances for use in our classification models. Fourth, we restrict our feature set to high-level natural language features that are more likely to generalize to classes on different topics rather than low-level domain-specific words. Finally, we design our models to generalize across teachers rather than optimizing to the speech patterns of individual teachers.

2. METHOD

2.1 Recording Teacher Audio

Data was collected at six rural Wisconsin middle schools during literature, language arts, and civics classes. Class sessions were taught by 11 teachers (three male; eight female) and lasted between 30 and 90 minutes. The teachers carried out their normal lesson plan, allowing the collection of a corpus of real-world samples of classrooms. Based on previous work [10], a Samson 77 Airline wireless microphone was chosen for teachers to wear while teaching. Teacher speech was captured and saved as a 16 kHz, 16-bit single channel audio file. A total of 37 class sessions were recorded on 17 separate days over a period of a year. The recordings contain a total of 32 hours and five minutes of audio.

2.2 Teacher Utterance Detection

Teacher speech was segmented into utterances using a voice activity detection (VAD) technique described in [14] and briefly reviewed here. Audio from the teacher's microphone was automatically split into potential utterances, consisting of either teacher speech or other sounds (e.g., accidental microphone contact, classroom noise), based on pauses (i.e., periods of silence) between speech. The beginning of a potential utterance was automatically identified when the amplitude envelope rose above a preset threshold. The end point of the utterance was automatically identified when the amplitude envelope dropped below this threshold for at least 1000 milliseconds, a pause of one second. The threshold was set to be sufficiently low so as to capture all instances of speech, also causing a high rate of false-alarms. False alarms were eliminated by filtering all potential utterances with Bing ASR [15]. If the ASR rejected a potential utterance, then it was discarded as a non-speech segment.

We validated the effectiveness of our VAD approach in an experiment by hand coding a random subset of 1,000 potential utterances as either containing speech or not containing speech [11]. We achieved an F_1 score of 0.97, which we deemed sufficiently accurate for the purposes of this study. Therefore, we applied our approach for VAD to the full dataset of 37 classroom recordings and extracted 10,080 utterances.

2.3 Question Coding and Transcription

We manually coded the complete set of automatically extracted utterances as containing a question or not. It should be noted that a known limitation of annotating automatically segmented speech is that each utterance may contain multiple tags (questions in this case), or conversely, a tag may be spread across over multiple utterances. This occurs because we use both a fixed amplitude envelope threshold and pause length to segment utterances, rather than creating specific thresholds for each teacher or class-session. This fully automates the VAD detection process, and allows us to test generalizability to new teachers. For this work, we allow question tags to span multiple utterances, since the entire content of question is likely to be essential to future work aimed at providing feedback to teachers.

We define a question after the question coding scheme developed by Nystrand and Gameron [4], [5], which is specific to classrooms. For example, calling on students in class (e.g., "What is the capital of Iowa [pause] Michael") is considered a question. Likewise, the teacher calling on a different student to answer the same question after evaluating the previous response (e.g., "Nope [pause] Shelby") is also considered a question. Calling a student name for other reasons, such as to discipline them, is not a question (e.g., "Steven"). Thus, question coding involves ascertaining both the context and intentionality of the utterance.

The coders were seven research assistants and researchers whose native language was English. Coders listened to the utterances in temporal order and assigned a label (question or not) to each based on the words spoken by the teacher, the teachers' tone (e.g., prosody, inflection), and the context of the previous utterance. Coders could also flag an utterance for review by a primary coder, although this occurred rarely.

As training, the coders first engaged in a task of labeling a common evaluation set of 100 utterances. These 100 utterances were selected to exemplify difficult cases. Once coding of the evaluation set was completed, the primary coder, who had considerable expertise with classroom discourse and who initially selected and coded the evaluation set, reviewed the codes. Coders were required to achieve a minimal level of agreement with the primary coder (Cohen's kappa, $\kappa = 0.80$). If the agreement was lower than 0.80, then errors were discussed with the coders.

After this training task was completed, the coders coded a subset of utterances from the complete dataset. In all, 36% of the 10,080 utterances were coded as containing questions. A random subset of 117 utterances from the full dataset were selected and coded by the expert coder. Overall the coders and the primary coder obtained an agreement of $\kappa = 0.85$ on this evaluation set.

From the full dataset of 10,080 labeled utterances, we selected a random (without replacement) subset of 1,000 utterances for manual transcription by humans. 30% of the utterances in this subset contained a question, which is slightly lower than the 36% question rate on the entire dataset.

2.4 Model Building

We trained and tested supervised classification models to predict if utterances contained part (or all) of a question, or did not contain a question. The model building process involved the following steps.

Features. Features were generated using the human transcripts for each utterance. We limited our feature set to a set of 37 generalizable NLP features to limit overfitting to teacher dialect or classroom subject/domain. These 34 features were obtained by processing each utterance with the Brill Tagger [16]. Each tagged token was examined for features (see [17] for further details) based on the semantics of various question types (e.g., causal, interpretation, disjunction) or the syntax of questions (e.g., WH-words and modal verbs). These 34 features capture key word (e.g., *why, how),* word categories (e.g., procedural), and parts of speech (e.g., noun, verb), and have previously been used to detect domain independent question properties associated with learning from human-transcribed questions [18]. Three additional features include proper nouns (e.g., student names), pronouns associated with teacher questions incorporating student responses (a type of question known as uptake), and pronouns not associated with uptake.

Minority oversampling. We supplemented *training* data with additional synthetic instances generated by the Synthetic Minority Over-sampling Technique (SMOTE) algorithm [19] in order to eliminate skew in the training set. Importantly, SMOTE was only applied to the training set and the original distributions in the testing set were not altered.

Classification and validation. We explored a number of classifiers: Naïve Bayes, logistic regression, random forest, J48 decision tree, J48 with Bagging, Bayesian network, *k*-nearest neighbor (k = 7, 9, and 11), and J48 decision tree, using implementations from the WEKA toolkit [20]. We also combined the classifiers with MetaCost, which penalized misclassifications of the minority class (weights of 2 and 4). All 37 features were used in the models.

We validated the classification models with leave-one-teacher-out cross-validation, in which models were built on data from 10 teachers (the training set) and validated on the held-out teacher (the testing set). The process was repeated for 11 folds so that each teacher appeared once in the testing set. This cross validation technique tests the potential of our models to generalize to new teachers in terms of variability in question asking and language.

3. RESULTS

The best performing model was Naïve Bayes, which achieved the overall highest F_1 score (0.53) for detecting utterances containing questions (the minority class). This model achieved an overall weighted F_1 score of 0.66 (see Table 1 for the confusion matrix).

Additionally, we also compared our results to a chance-model that assigned the question label at the same rate as our model, but did so randomly. We calculated the chance recall and precision for the question label as the average value per teacher over 10,000 iterations. We consider this approach to computing chance to be more informative than a naïve minority baseline model that would yield perfect recall but negligible precision. We observed an encouraging level of recall (0.61) for the question class, which reflects the model's ability to detect questions from utterances well above both chance precision (0.32) and recall (0.42). However, we note that further refinement is needed to improve the model's precision (0.47), which is hindered by the frequent misclassification of utterances as questions.

Table 1. Confusion matrix of 1,000 utterance subset, showing the count and the proportion in parenthesis.

Instances	Actual	Predicted	
		Question	*Utterance*
320	*Question*	195 (0.61)	125 (0.39)
680	*Utterance*	224 (0.33)	456 (0.67)

4. GENERAL DISCUSSION

Questions play a central role in dialogic instruction in classrooms. The importance of dialogue and discussion is widely acknowledged in research [6], [9], [20] and public policy (e.g., Common Core State Standards for Speaking and Listening). The ability to automatically detect questions for both research and teacher professional development might have important consequences in improving student engagement. Towards this goal, our current work focuses on semi-automatic prediction of individual teacher questions teacher audio recorded in live classrooms.

We demonstrated promising results with our approach, consisting of manually transcribed automatically segmented teacher speech, high-level language features, and machine learning. Our best model, validated independently of the teacher, achieved an overall F_1 score of 0.66 and a F_1 score for the question class of 0.53. This reflects a modest improvement in overall classification (F_1 of 0.63) and a significant improvement in question detection accuracy (F_1 of 0.40) over a recent state of the art model [13].

A major contribution of our work is that our models were trained and tested only on automatically, and thus imperfectly, segmented utterances. This confirms that question detection on imperfect sentence boundaries is possible, a result that furthers the work of [13], in which the authors split a subset of manually defined sentences on the longest silence in the sentence (see Section 1.1).

Despite these encouraging results, this study is not without limitations. Most importantly, we only considered manually transcribed speech in order to examine the feasibility of the automatic identification of questions derived from noisy classroom environments. To fully automate our approach we will need to incorporate ASR engines. We expect that the incorporation of noisy ASR will contribute to additional errors in classification, a possibility we are studying in ongoing work that applies automatic speech recognition (ASR) on our full dataset of 10,080 utterances.

Research [11]–[13] indicates that acoustic and contextual features may be important to capture certain difficult types of questions and we will explore the use of these features in future work. Furthermore, additional data collection which includes a second microphone that captures general classroom activity is ongoing. This second channel of audio, when combined with the recording of the teacher, will allow modelling patterns of teacher-student interactions, potentially revealing question-response patterns between teachers and students. Finally, we will extend our approach to classify the question properties defined by Nystrand and Gameron [9]. We have previously explored this task using human transcriptions of manually segmented questions [18], [21], but will extend this work using our approach that employs automatic segmentation and subsequently ASR transcriptions.

In summary, we took steps towards fully automating the detection of teacher questions from audio recordings of live classrooms. We will continue to refine and improve these models as we extend our approach to use ASR transcriptions of the utterances. The present contribution is one component of a broader effort to automate the collection and coding of classroom discourse to improve learning. The automated system is intended to generate personalized formative feedback to teachers, enabling reflection and improvement of their pedagogy, with the ultimate goal of increasing student engagement and achievement.

5. ACKNOWLEDGMENTS

This research was supported by the Institute of Education Sciences (IES) (R305A130030). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the author and do not represent the views of the IES.

6. REFERENCES

[1] S. Kelly, "Classroom discourse and the distribution of student engagement," *Soc. Psychol. Educ.*, vol. 10, no. 3, pp. 331–352, 2007.

[2] W. Sweigart, "Classroom talk, knowledge development, and writing," *Res. Teach. Engl.*, vol. 25, no. 4, pp. 469–496, Dec. 1991.

[3] M. K. Lai and S. McNaughton, "Analysis and discussion of classroom and achievement data to raise student achievement," in *Data-based decision making in education*, Springer, 2013, pp. 23–47.

[4] M. Nystrand, A. Gamoran, R. Kachur, and C. Prendergast, "Opening dialogue," *Teach. Coll. Columbia Univ. N. Y. Lond.*, 1997.

[5] A. Gamoran and S. Kelly, "Tracking, instruction, and unequal literacy in secondary school english," *Stab. Change Am. Educ. Struct. Process Outcomes*, pp. 109–126, 2003.

[6] A. N. Applebee, J. A. Langer, M. Nystrand, and A. Gamoran, "Discussion-Based Approaches to Developing Understanding: Classroom Instruction and Student Performance in Middle and High School English," *Am. Educ. Res. J.*, vol. 40, no. 3, pp. 685–730, 2003.

[7] M. Nystrand, "Research on the role of classroom discourse as it affects reading comprehension," *Res. Teach. Engl.*, vol. 40, no. 4, pp. 392–412, May 2006.

[8] M. Nystrand and A. Gamoran, "Instructional discourse, student engagement, and literature achievement," *Res. Teach. Engl.*, pp. 261–290, 1991.

[9] M. Nystrand, L. L. Wu, A. Gamoran, S. Zeiser, and D. A. Long, "Questions in time: Investigating the structure and dynamics of unfolding classroom discourse," *Discourse Process.*, vol. 35, no. 2, pp. 135–198, 2003.

[10] S. K. D'Mello, A. M. Olney, N. Blanchard, B. Samei, X. Sun, B. Ward, and S. Kelly, "Multimodal capture of teacher-student interactions for automated dialogic analysis in live classrooms," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, New York, NY, USA, 2015, pp. 557–566.

[11] K. Boakye, B. Favre, and D. Hakkani-Tur, "Any questions? Automatic question detection in meetings," in *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, 2009, pp. 485–489.

[12] A. Stolcke, N. Coccaro, R. Bates, P. Taylor, C. Van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer, "Dialogue act modeling for automatic tagging and recognition of conversational speech," *Comput. Linguist.*, vol. 26, no. 3, pp. 339–373, 2000.

[13] L. Orosanu and D. Jouvet, "Detection of sentence modality on French automatic speech-to-text transcriptions," in *International Conference on Natural Language and Speech Processing*, Alger, Algeria, 2015.

[14] N. Blanchard, M. Brady, A. M. Olney, M. Glaus, X. Sun, M. Nystrand, B. Samei, S. Kelly, and S. D'Mello, "A study of automatic speech recognition in noisy classroom environments for automated dialog analysis," in *Artificial Intelligence in Education*, 2015, pp. 23–33.

[15] Microsoft, "The Bing Speech Recognition Control," May 2014.

[16] E. Brill, "A simple rule-based part of speech tagger," in *Proceedings of the Workshop on Speech and Natural Language*, 1992, pp. 112–116.

[17] A. Olney, M. Louwerse, E. Matthews, J. Marineau, H. Hite-Mitchell, and A. Graesser, "Utterance classification in AutoTutor," in *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing Volume 2*, 2003, pp. 1–8.

[18] Samei, B., Olney, A. M., Kelly, S., Nystrand, M., D'Mello, S., Blanchard, N., Sun, X., Glaus, M. & Graesser, A., "Domain independent assessment of dialogic properties of classroom discourse," in *7th International Conference on Educational Data Mining*, 2014, pp. 233–236.

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, pp. 321–357, 2011.

[20] National Governors Association Center for Best Practices and Council of Chief State School Officers, "Common Core State Standards Speaking & Listening." National Governors Association Center for Best Practices, Council of Chief State School Officers, Washington D.C., 2010.

[21] Samei, Borhan, Olney, Andrew M., Kelly, Sean, Nystrand, Martin, D'Mello, Sidney, Blanchard, Nathaniel, and Graesser, Art, "Modeling classroom discourse: Do models of predicting dialogic instruction properties generalize across populations?," in *Proceedings of the Eighth International Conference on Educational Data Mining*, Madrid, Spain, 2015, pp. 444–447.

Modeling Interactions Across Skills: A Method to Construct and Compare Models Predicting the Existence of Skill Relationships

Anthony F. Botelho
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609-2280
abotelho@wpi.edu

Seth A. Adjei
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609-2280
saadjei@wpi.edu

Neil T. Heffernan
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609-2280
nth@wpi.edu

ABSTRACT

The incorporation of prerequisite skill structures into educational systems helps to identify the order in which concepts should be presented to students to optimize student achievement. Many skills have a causal relationship in which one skill must be presented before another, indicating a strong skill relationship. Knowing this relationship can help to predict student performance and identify prerequisite arches. Skill relationships, however, are not directly measurable; instead, the relationship can be estimated by observing differences of student performance across skills. Such methods of estimation, however, seem to lack a baseline model to compare their effectiveness. If two methods of estimating the existence of a relationship yield two different values, which is the more accurate result? In this work, we propose a method of comparing models that attempt to measure the strength of skill relationships. With this method, we begin to identify those student-level covariates that provide the most accurate models predicting the existence of skill relationships. Focusing on interactions of performance across skills, we use our method to construct models to predict the existence of five strongly-related and five simulated poorly-related skill pairs. Our method is able to evaluate several models that distinguish these differences with significant accuracy gains over a null model, and provides the means to identify that interactions of student mastery provide the most significant contributions to these gains in our analysis.

Keywords

prerequisite structures, skill relationships, feature selection, model comparison

1. INTRODUCTION

Many educational systems like ASSISTments and Khan Academy already implement a prerequisite structure as a suggested ordering in which skills should be presented to students. These structures are often developed by domain experts and teachers in the field of study, and are likely to hold ground-truth. It is clear, for example, that relationships can be identified by observing skills at the problem-level; by viewing the steps required for students to complete each item, it can be known that any skills required to complete such problems can be considered prerequisites. For example, Multiplying Whole Numbers may act as a prerequisite to Greatest Common Factors, as is used in our analysis. While causality suggests a strong relationship, it is possible for two skills to relate to each other in other ways. Such relationships are less intuitive, perhaps requiring a similar thought process or sequence of steps to solve, even if the content of such tasks differ. Many causal skill arches are identifiable by domain experts by observing content, but as described, other such relationships may be missed due to their non-intuitive structures. By observing strong skill relationships identified by domain experts, we construct a method of measuring the factors that are most predictive of their existence.

We also argue that identifying strong relationships is not enough for a method of prediction to be considered adequate. Such a method should also be able to identify weak or non-existent skill relationships. It is likely that while much attention and research is placed on structuring prerequisite links, some of these are false-positives. In other words, a skill may be listed as a prerequisite, but has no true relationship to its supposed post-requisite skill. In such a case there is little or no interactions of performance. Such links must also be identified and removed or reordered in learning platforms to benefit the students.

A significant amount of research has looked at measuring the strength of skill relationships [1],[4], and even the effects such relationships have on measuring student performance [3],[10], but without understood ground truths, it is difficult to compare across these methods. Furthermore, many of these methods represent similar conceptualizations of performance inherently, or through variations of representation such as aggregation or centering. For example, "student achievement" is likely a predictor of skill relationships (achievement on a prerequisite skill will likely influence achievement on a post-requisite skill), but can be represented as the percent of problems answered correctly, mastery speed (the number of items needed to complete an assignment as is commonly used in intelligent tutoring systems), or countless other combinations of features. It will be

important to distinguish between these generalized components to avoid incorporating features that capture the same types of conceptualizations into predictive models.

This work provides a method to evaluate models that measure the strength of skill relationships, and with this model we attempt to identify which features best indicate a strong relationship between two skills. This analysis will incorporate a method of generalizing and distinguishing features that measure different aspects of learning and performance. With this methodology, we seek to answer the following two research questions:

1. What link-level features, expressed in this paper as interactions of performance between skills, are significant in predicting the existence or non-existence of skill relationships?

2. Which features are the strongest predictors of skill relationships, and does combining them make for a more accurate predictive model?

The next section of this paper will discuss some of the previous research performed on skill relationships and prerequisite structures. Then, we will discuss our theory and methodology to provide a baseline model of comparing methods of measuring skill relationships. Using this model, we then compare several commonly-used student-level features, and of the most accurate, compare several different representations of those features. Finally, we will discuss our findings and suggested future works.

2. PREVIOUS WORKS

The discovery and refinement of prerequisite skill structures has been an important research question in recent years. The impact of this research on educational systems cannot be overemphasized. Domain experts who design these structures need data centered methods to support the decisions they make; it is vital to have empirical data to support hypothesis regarding the order in which skills are presented as it can have a large impact on student achievement and either aid or impede the learning process. Additionally, identifying the best prerequisite skill structure will enhance student modeling; knowing a student's prior performance on prerequisite skills can help estimate that student's performance on the post-requisites. This can lead to earlier interventions for struggling students, or even help redefine mastery perhaps students who perform very well on a prerequisite requires less practice on a post-requisite, or can be given more advanced examples.

Tatsuoka, defined a data structure called the Q-Matrix, that represents the mapping of problems to skills: the rows of this matrix represent the problems, and the columns represent the skills [9]. Though the goal of the research was to diagnose the misconceptions of students, they set in motion a number of studies that have used this data structure as the first step to find prerequisite structures [2],[5],[8].

Desmarais and his colleagues developed an algorithm that finds the prerequisite relationship between questions, or items, in students' response data [6]. They compare pairs of items in a test and determine any interactions existing between each pair. Depending on the interactions and a set of interaction-related criteria, they determine whether the two items have a prerequisite relationship between them. This approach was applied by Pavlick, et al. to analyze item-type covariances and to propose a hierarchical agglomerative clustering method to refine the tagging of items to skills [7]. Brunskel conducted a preliminary study in which they use students' noisy data to infer prerequisite structures [4]. Further research by Scheines, et al. extended a causal structure discovery algorithm in which an assumption regarding the purity of items is relaxed to reflect real data and to use that to infer prerequisite skill structure from data [8].

3. DATASET

The dataset[1] used for this study consists of real-world student data from the ASSISTments online learning platform. The raw data contains student problem logs pertaining to ten math skills from the 2014-2015 school year. These ten skills represent five skill pairs, listed in Table 1, for which domain experts identified as having a strong prerequisite relationship. While we are not limiting the usage of our proposed baseline model to just prerequisite relationships, these are the most reliable to identify due to the causal effect of content (if problems in skill B require the use of skill A to complete, a strong relationship can be identified).

Table 1: The strong skill pairs as determined by domain experts

Prerequisite	Post-requisite
Multiplication of Whole Numbers	Greatest Greatest Common Factor
Subtracting Integers	Order of Operations
Division of Whole Numbers	Dividing Multi-Digit Numbers
Volume of Rectangular Prisms Without Formula	Volume of Rectangular Prisms
Nets of 3D Figures	Surface Area of Rectangular Prisms

In order to identify believable ground-truth skill pairs, a survey containing 24 skill pairs for which we had sufficient student data (greater than 50 student rows) was administered to 45 teachers and domain experts who use ASSISTments. Each was asked to rate on a scale of 1 to 7, indicating the perceived qualitative strength of the relationship of each skill pair. From the survey results, five skill pairs were selected to be the strongest related links with the smallest variance in opinion scores. As we are treating these links as truth, we wanted to be highly selective of these pairs.

The resulting dataset consists of 1838 total student rows from 896 unique students. This includes two rows of data per student for each of the five skill pairs included. The first row contains information of that student's performance on the pre- and post-requisite skills, while the second row contains student performance on the prerequisite and a simulated post-requisite described further in the next section.

[1]The full raw and filtered datasets are available at the following link: http://tiny.cc/veqg5x

For each student, a feature vector was selected using common performance metrics to compare within our model. This feature vector contained eight link-level features representing the interactions between student-level prerequisite and post-requisite performance metrics. The generated link-level features observed are as described below:

Percent Correct
The mean-centered[2] percentage of correct responses in the prerequisite skill multiplied by the mean-centered percentage of correct responses in the post-requisite skill.

First Problem Correctness (FPC)
The binary correctness of the first response in the prerequisite skill multiplied by the binary correctness of the first response on the post-requisite skill.

Mastery Speed
The mean-centered mastery speed of the prerequisite skill, defined as the number of problems required for each student to achieve three consecutive correct responses, multiplied by the mean-centered mastery speed of the post-requisite skill. In addition to centering, these values were also winsorized to make the largest possible value 10, chosen as this is often the maximum number of daily attempts allowed within ASSISTments. All centering and winsorizing occurred before multiplying the two values.

Z-Scored Percent Correct
The z-scored[3] value of mean-centered percentage of correct responses in the prerequisite skill multiplied by the z-scored value of mean-centered percentage of correct responses in the post-requisite skill.

Binned Mastery Speed (Bin)
The numbered bin of mastery speed as described in [3] of the prerequisite skill multiplied by the bin of mastery speed in the second skill. Students were placed into one of five bins based on mastery speed if the assignment was completed and based on percent correct if the assignment was not completed.

Z-Scored Mastery Speed
The z-scored value of mean-centered, winsorized mastery speed in the prerequisite skill, multiplied by the z-scored value of mean-centered, winsorized mastery speed in the post-requisite skill.

Bin X FPC
The binned mastery speed value in the prerequisite skill multiplied by the binary correctness of the first response in the post-requisite skill.

Percent Correct X FPC
The mean-centered percentage of correct responses in the prerequisite skill multiplied by the binary correctness of the first response in the post-requisite skill.

[2]All centering of features was performed at the skill-level.
[3]All z-scoring was performed at the class-level.

4. METHODOLOGY

The ultimate goal of this work is to provide the means of comparing models predicting the existence, or non-existence of skill relationships. Our approach to this is through the comparison and identification of features that most accurately predict these relationships. Using principal component analysis, we group similar features into more generalized conceptualizations to both compare which types of features matter when predicting relationships, but also to avoid problems of multicollinearity that may bias our estimates. Once this baseline model is established, we can construct new predictive models from the significant features and observe their accuracy in predicting the existence of skill relationships when compared to a simple null, or unconditional model.

In order to compare the usage of features against a weak or non-existent relationship, we simulated a new skill using students from the existing prerequisite skill by generating random sequences of responses. For each existing student, we randomly assign him/her a probability between 0.5 and 0.9 in order to create a random sequence of answers. For example, a student given a probability of 0.5 has a 50% chance of answering each given problem correctly. We simulate student answers until either mastery is achieved, defined as three sequentially correct responses, or the student reaches 10 problems without mastering; a value of 10 is chosen here, as many assignments in ASSISTments are given a daily limit of 10 problem attempts before asking the student to seek help or try again on another day. While we acknowledge there are many ways to accomplish this simulation step, we feel this simple method sufficiently creates a skill that has no relationship to the original prerequisite as intended. As our proposed method is intended to be used in the future to help identify undiscovered pre- or post-requisite links, we chose to use a simulated skill rather than a random existing skill to avoid the possibility of randomly selecting an undiscovered related skill. Again, we wanted to be highly selective and consider several such scenarios as we are attempting to create ground-truth values to which we can make our comparisons.

Using these two skill-pairs, one link representing a strong relationship while the other representing a non-existent relationship, we can calculate a feature vector for each student in the prerequisite skill with values from each skill-pair. We use a binary logistic regression with the existence of a relationship as the dependent variable and several link-level covariates to predict whether a skill relationship exists for each student row. The existence of a relationship can be determined then simply by majority ruling, but such calculation is not included in this work and instead observes accuracy at the student-level for a more accurate comparison.

We begin to compare commonly used student-level features in this study through two levels analysis. The first step attempts to compare groups of features, generalizing different representations of similar features into conceptual groupings. As such, we are able to view the predictive power of what we denote as initial performance, mastery, and correctness. The second experiment looks at the individual features as different representations of the overall group to compare

	Component		
	1	2	3
Percent Correct		.821	
First Problem Correctness (FPC)			.839
Mastery Speed	.969		
Z-Scored Percent Correct		.865	
Binned Mastery Speed (Bin)	.972		
Z-Scored Mastery Speed			
Bin X FPC			.873
Percent Correct X FPC		.612	

Figure 1: The results of the PCA analysis. All features except Z-Scored Mastery Speed mapped to one of three generalized components.

these predictors at a closer level. We can take each factor of mastery, for example, and compare their usage in several models to determine which is the most accurate predictor of the existence of skill relationships.

4.1 Comparing Link-Level Features

In order to compare representations of student-level features, we must first be able to compare general conceptualizations of features to determine which provide more accurate predictions of the existence of skill relationships. We want to capture the true representations of each metric and attempt to interpret these generalizations as types of features. In order to accomplish this grouping of predictors, we use principal component analysis (PCA) to identify which student-level features correlate to and are representative of more generalized components. PCA is primarily used for dimensionality reduction as we are doing here and gives us the ability to create new variables from the component mappings. The resulting feature alignment can be seen in Figure 1. As is the case in our study, and was mentioned in the previous section, we have multiple metrics of mastery speed as well as several other features. As we can represent "mastery" in several ways, we want to know if the overall concept of mastery, as captured by the metrics used, is reliably predictive of the existence of skill relationships.

Creating a new set of predictors of these groupings, we are able to incorporate these into a binary logistic regression model to view the predictive power of each. While PCA groups similar features together based on their correlations, by viewing which features are grouped we are able to interpret and label each. From this process, we found that most of our features fell into three categories for which we have given the names "mastery," as this consists of representations of mastery speed, "correctness," as this consists of representations of the percentage of correct student responses, and "initial performance," as this consists of representations of student performance on the initial items of each skill. In addition to these three categories, we are also left with student mastery speed z-scored within student classes as a variable that did not fall under either of the three aforementioned categories; while a derivation of mastery speed, we believe that this did not correlate to the "mastery" category due to the method of standardization as it is capturing this metric in relation to students' peers. We will readdress this case in our section of discussion.

Once these predictors are identified and created, we construct a binary logistic regression model to predict, for each student row, whether a relationship exists or not. This model will give us a significance value and coefficient for each predictor in the model, as well as an overall predictive accuracy of the model which will be used more for the next analysis.

4.2 Comparing Feature Models

After being able to compare which generalized groups of features are significant predictors of the existence of skill relationships, we are able to compare the individual student-level features that fall into each category by incorporating them into separate models to observe predictive accuracy. The analysis of the first experiment is used to determine which categories are significant in predicting the existence of skill relationships. Using that information, we are able to focus on those groupings with significance to construct models that utilize factors from each grouping. The grouping of "mastery," for example contains the factors of mastery speed and binned mastery speed, so we can construct models using each to compare differences in predictive power. To avoid problems of collinearity, no single model contains more than one factor from a single grouping. This significantly reduces the number of combinations of features to test compared to running this experiment without first grouping like features and identifying those that are significant as we did in the first experiment.

Using the significant groupings, we are able to create 17 models consisting of single, pairs, and triplets of features. A logistic regression is run on each of these models to predict the existence of a skill relationship. Of the 17 models, 10 of them produce a statistically significant prediction when compared to a null model. Ideally, our null model should produce a 50% accuracy as there is an equal number of good and bad link rows in our dataset. This is not always the case, however, as depending on the feature observed, information may be missing for a particular student; mastery speed, for example, as the number of items attempted by a student before reaching 3 consecutive correct answers, would be missing for any student that did not complete the assignment. For this reason, the predictive power of each model is described as gains in predictive accuracy, or rather, the accuracy of each model minus the accuracy of the corresponding null model.

5. RESULTS

The results of the first analysis are expressed in Table 2. Each of the three feature groupings of Mastery, Correctness, and Initial Performance created using PCA in addition to the Z-Scored Mastery are compared within the same model, predicting the existence of a skill relationship. As these

Table 2: The coefficients and significance values of the generalized components analyzed. From this we can focus on models that exclude features contained in the components with no significance.

Component	Coefficient Value (log-odds units)	Significance
Mastery	-.251	<.001***
Correctness	.015	.802
Initial Performance	.129	.037*
Z-Scored Mastery Speed	-.129	<.001***

again are link-level features describing interactions between student-level performance on prerequisite and post-requisite skills, it is difficult to draw tangible interpretations from the coefficient value, expressed in log-odds units. This coefficient, used in the logistic regression to make the predictions, describes each component's effect on the dependent variable. For example, for each unit increase in "Mastery," the probability that the link exists decreases. Again, as this component is an aggregation of interaction features, it is really describing an aggregation of differences of differences between student-level features making it difficult to make definitive claims regarding these values alone and were included purely to display a general trend of these components on the prediction.

From the table, we are able to determine the significance of each component on the overall prediction by viewing the corresponding p-values in the third column. Looking at these values, we can claim that the overall grouping of "Correctness" seems to have less of an impact on the predictive accuracy of the model. As this term is not significant, we can focus the remainder of our study on the remaining three components.

Table 3 illustrates the results of our second analysis comparing the models that we are able to construct with the remaining features once the "Correctness" grouping has been disregarded. This figure shows the comparative predictive accuracy of the 10 models that give statistically significant predictions as seen in Table 3. Again, these values are expressed as accuracy gains, or rather the percent accuracy increase over the null model run for each predictive model.

6. DISCUSSION

This work provides a baseline model of comparing student-level performance across skills to measure the strength of a skill relationship and compare the accuracy of both features and models that estimate this value. Such a model, in our experience, has not existed prior to this study. Our method attempts to identify not only the individual features that contribute to better predictions of these relationships, but also moves to generalize similar features into conceptualizations for comparison in order to minimize multicollinearity.

The principal component analysis step of our model found that all but one feature mapped to one of three components that we have interpreted as mastery, correctness, and initial performance. It was found the z-scored mastery speed, contrary to our intuition, did not map well to the grouping of mastery. We can speculate the reason for this occurrence by altering our interpretation of the feature. Mastery speed itself is an interesting metric as it attempts to capture two dimensions of performance: a level of understanding and a rate of learning. Also, to reiterate a prior distinction, these metrics are interactions of performance across skills. By z-scoring the metric, it is capturing a contextual effect of each student in comparison with other students in the class, a distinction that appears to have a significant effect.

Observing the resulting model components from the principal component analysis in Table 2, we were able to focus our attention to those components with significant values. Correctness was the only component of that model that was found to have no statistical significance on the dependent variable. This is certainly interesting, as percent correctness and other such measures are among the most common metrics of performance. Perhaps the interaction between pre- and post-requisite percent correct is losing some predictive power from when the metric is used for other predictions of performance.

This aspect illustrates one other important finding that the distinct representations of one metric or another each contribute differently to the predictive accuracy of the models studied. Models incorporating mastery speed, for example, had no significant accuracy gains over a null model, while mastery speed binning showed considerable gains as seen in Table 3. The baseline model of comparison proposed in this study provides the means to make that distinction regarding features contained within the same generalized component grouping. As is seen in that figure, combinations of features outperform any single feature, illustrating a more robust model by capturing multiple representations of performance.

7. FUTURE WORK

While we have shown that our model is able to compare and identify features that contribute to higher accuracy in predicting the existence of skill relationships, we also need to stress the importance of the usage of this information. The ability to compare features is only the first step of our model's goal. By identifying strong predictors of skill relationships that we know exist, we can apply it to other skills within ASSISTments and other systems to identify potentially new prerequisite arches, and also to better measure and predict long-term student performance, learning, and retention. Having an accurate estimate of skill relationships can help restructure prerequisite structures to provide skill sequences in an order that optimizes student learning and achievement.

The work in this paper incorporated several skills into a single dataset to make predictions. In this case, we wanted to create a method that is generalizable to some degree. While our selective skill set allows us to make some claims in terms of the accuracy these models over all skills, it may likely be the case that skill relationships are measurable in different ways for different skills. Further analysis could repeat the steps here on each one of the acquired skills in the dataset.

Table 3: The models constructed from features in the significant generalized components. No one model contains more than a single feature from each generalized component.

Model	Null Accuracy	Model Accuracy	Accuracy Gain	Significance
Mastery Speed (MS)	0.63	0.62	0.00	1.000
Z-Scored Mastery Speed	0.63	0.63	0.00	0.888
First Problem Correctness (FPC)	0.50	0.56	0.06	<0.001***
Binned MS	0.50	0.69	0.19	<0.001***
Bin X FPC	0.50	0.56	0.06	<0.001***
Bin, Z-Scored MS	0.50	0.71	0.21	<0.001***
MS, FPC	0.63	0.62	0.00	1.000
MS, Bin X FPC	0.63	0.62	0.00	1.000
Bin, FPC	0.50	0.69	0.19	<0.001***
Bin,Bin X FPC	0.50	0.69	0.19	<0.001***
MS, FPC, Z-Scored MS	0.63	0.63	0.00	0.754
MS, Bin X FPC, Z-Scored MS	0.63	0.63	0.00	0.979
Bin, FPC, Z-Scored MS	0.50	0.71	0.20	<0.001***
Bin, Bin X FPC, Z-Scored MS	0.50	0.71	0.21	<0.001***
MS, Z-Scored MS	0.63	0.63	0.00	0.843
FPC, Z-Scored MS	0.50	0.64	0.14	<0.001***
Bin X FPC, Z-Scored MS	0.50	0.61	0.11	<0.001***

While correctness was not significant in these results, perhaps it is significant when predicting certain types of skills. Perhaps, similar to our features, skills themselves could be generalized into conceptual types for different kinds of analysis pertaining to interactions of performance and their relationships.

The feature vectors generated for each student in our dataset captured many of the most common student-level metrics, but certainly not all of them. There are many other aspects that could be added including completion, measures of learning rate, time spent on the assignments, hint usage, and countless other variables. In addition, this study only observed interactions expressed as multiplications of these terms to describe them as link-level features. There are various other ways to represent interactions or other such transformations including differences of values, division of values, or just simply cross-feature interactions as was partially explored here by looking at Bin X FPC and Percent Correct X FPC. Such interactions model various other aspects of student performance and behavior that can be very useful in this type of relationship prediction.

The methodology presented observes models that predict the existence of skills as a binary outcome, while it can be modified to make comparisons on estimates of relationship strengths as a continuous outcome as well. The method observed model accuracy at the student level for better measurements, but it is a skill-level relationship that is being tested. One simple addition of future work could explore how to best combine the predictions at a student level to make a skill-level prediction. The methodology can then test relationships on the entire system skill structure.

8. ACKNOWLEDGMENTS

We acknowledge funding from the NSF (1316736, 1252297, 1109483, 1031398, 0742503, 1440753), the U.S. Dept. of Ed. GAAN (P200A120238), ONR's "STEM Grand Challenges," and IES (R305A120125, R305C100024).

9. REFERENCES

[1] S. Adjei, D. Selent, N. Heffernan, Z. Pardos, A. Broaddus, and N. Kingston. Refining learning maps with data fitting techniques: Searching for better fitting learning maps. In *Educational Data Mining*, 2014.

[2] T. Barnes. The q-matrix method: Mining student response data for knowledge. In *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop*, 2005.

[3] A. Botelho, H. Wan, and N. Heffernan. The prediction of student first response using prerequisite skills. In *Learning at Scale*, 2015.

[4] E. Brunskill. Estimating prerequisite structure from noisy data. In *Educational Data Mining*, pages 217–222. Citeseer, 2011.

[5] Y. Chen, P.-H. Wuillemin, and J.-M. Labat. Discovering prerequisite structure of skills through probabilistic association rules mining. In *The 8th International Conference on Educational Data Mining*, pages 117–124, 2015.

[6] M. C. Desmarais, A. Maluf, and J. Liu. User-expertise modeling with empirically derived probabilistic implication networks. *User modeling and user-adapted interaction*, 5(3-4):283–315, 1995.

[7] P. I. Pavlik Jr, H. Cen, L. Wu, and K. R. Koedinger. Using item-type performance covariance to improve the skill model of an existing tutor. *Online Submission*, 2008.

[8] R. Scheines, E. Silver, and I. Goldin. Discovering prerequisite relationships among knowledge components. In *Educational Data Mining*, 2014.

[9] K. K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of educational measurement*, 20(4), 1983.

[10] H. Wan and J. B. Beck. Considering the influence of prerequisite performance on wheel spinning. In *Educational Data Mining*, 2015.

Robust Predictive Models on MOOCs : Transferring Knowledge across Courses

ABSTRACT

As MOOCs become a major player in modern education, questions about how to improve their effectiveness and reach are of increasing importance. If machine learning and predictive analytics techniques promise to help teachers and MOOC providers customize the learning experience for students, differences between platforms, courses and iterations pose specific challenges. In this paper, we develop a framework to define classification problems across courses, provide proof that ensembling methods allow for the development of high-performing predictive models, and show that these techniques can be used across platforms, as well as across courses. We thus build a universal framework to deploy predictive models on MOOCs and demonstrate our case on the dropout prediction problem.

Keywords

Transfer Learning, Ensembling methods, Stacking, MOOCs, Dropout prediction

1. INTRODUCTION

As Massive Open Online Courses (MOOCs) continue to become a vital part of modern education, it becomes more and more necessary to increase their effectiveness and reach. Along with learning science and design, data analytics is known to be one of the fields most likely to improve this new education experience ([1]). Predictive analytics are particularly promising, allowing researchers to design real-time interventions and to adapt course content based on student behavior ([7],[9],[6],[3]).

Ideally, these predictive analytics would act in ways similar to an experienced teacherâĂŤone who is able to identify different students, and to adapt her actions accordingly. However, because the data available for training models is often significantly different then the data to which those models will be applied, it can be challenging to fully realize this promise ([8]). A predictive analytics system for MOOCs should be able to build on accumulated "past data" to make accurate predictions about an ongoing class. Thanks to the vast offerings of MOOC databases like edX and Coursera, there is now a plethora of past data available, both across and within a given course.

But this diversity of available courses also means the goal of real-time prediction is easier set than accomplished. Courses may come from different platforms, focus on different topics, or occur at different times. They may have more or less homework, span different lengths of time, or require different levels of involvement. As platforms evolve, courses may also morph to include new information or fulfill shifting demands. Such changes typically affect the behavior of students.

This raises a number of questions and challenges for a data or learning scientist. Given data from a set of repeatedly offered MOOC courses, key questions that shape the design of relevant predictive analytics methods are as follows:

Purpose Can I use past courses to predict outcomes within an ongoing course?

Data What data should I exploit to build my predictions? Is data from a single course enough, or should I use several courses?

Method What method will achieve good efficacy if I use data from a single course? Several courses?

In this paper, we address the challenges inherent in building predictive models that perform well across courses. We answer the questions, mentioned above, that a MOOC analyst would ask about the prediction objectives, the data, and the methods used to build such models. We also address whether such methods are able to perform well across courses on the same platform, and on different platforms.

This paper is divided into five sections. The remainder of this first section explores the available literature regarding MOOC dropout prediction and ensembling methods in machine learning. Section 2 introduces the formal notations, assumptions, and data sets we used to prove our case. Section 3 details different methods that prove useful for building robust models that transfer well to new courses. Section 4 presents the evaluation metrics, and showcases the effectiveness our techniques on the dropout prediction problem. Section 6 evaluates the potential impact of such techniques, and summarizes the key findings and 7 conclusions.

Literature review

Even before the recent e-learning boom, concerned researchers

have attempted to predict dropout. One major obstacle facing such attempts is the difficulty of building robust predictive algorithms. While working with early e-learning data, the authors of [7] improved the performance of their learning algorithm by merging several predictive algorithms together, namely Support Vector Machines, Neural Networks, and Boosted Trees.

Since then, almost all dropout studies have been conducted on MOOC data. Some researchers (like the authors of [9], who studied the effects of collaboration on the dropout rate of students) focus on understanding drivers of dropout among students. Others develop feature extraction processes and algorithms capable of pinpointing at-risk students before they drop out. If a MOOC is able to identify such students early enough, these researchers reason, it may be possible for educators to intervene. In [6], Halawa et. al. used basic activity features and respective performance comparison to predict dropout one week in advance. The authors of [2] included more features, as well as an integrated framework that allowed users to apply these predictive techniques to MOOC courses from various eligible platforms.

As MOOC offerings proliferate, the ability to "transfer" statistical knowledge between courses is increasingly crucial, especially if one wants to predict dropout in real time. Unfortunately, it is often difficult to take models built on past courses and apply them to new ones. In [3], Boyer and Veeramachaneni showed that models built on past courses don't always yield good predictive performance when applied to new courses.

Because there is generally only one dataset available per course, the ability to build robust models on MOOCs has naturally accompanied the rise of ensemble methods. Over the past twenty years, a flourishing predictive literature has appeared, offering various techniques for choosing and ensembling models in order to achieve high-performing predictors. A technique called "stacking" has proven particularly promising. In [5], Svzeroski et. al. showed that stacking models usually perform as well as the best classifiers. They also confirmed that linear regression is well-suited to learning the metamodel, and introduced a novel approach based on tree models. The authors of [4] demonstrated the possibility of incrementally adding models to the "ensembling base" from a pool of thousands. Sakkis et. al. [10] used the stacking method to solve spam filtering problems, finding that it significantly improved performance over the benchmark.

In this paper, we explore a framework conducive to building robust predictive models applicable to MOOCs. Although we do address dropout prediction specifically, we also consider the broader possibilities for building predictive models from a set of courses.

2. PROBLEM SETTING AND DATA SETS

We place ourselves in the context of using past courses to build a predictive model for a unseen course. We use the term *source* courses for those courses whose data is used to build (train) the predictive models, and the term *target* course for the initially "unseen" course. We consider the general problem of predicting for each student i an outcome y_i^t at time t in the future. We suppose that we have access to information about each student's behavior through a set of features: for example, a behavioral vector $x_i^t \in R^d$ describes the behavior of the student i at time t.

ID	Name	Platform	Students	Weeks
C_0	6002x13	edX	29,050	14
C_1	6002x12	edX	51,394	14
C_2	201x13	edX	12,243	9
C_3	3091x12	edX	24,493	12
C_4	3091x13	edX	12,276	14
C_5	aiplan_001	Coursera	9,010	5
C_6	aiplan_002	Coursera	6,608	5
C_7	aiplan_003	Coursera	5,408	5
C_8	animal_001	Coursera	8,577	5
C_9	animal_002	Coursera	5,431	5
C_{10}	astrotech_001	Coursera	6,251	6
C_{11}	codeyourself_001	Coursera	9,338	7
C_{12}	criticalthinking_1	Coursera	24,707	5
C_{13}	criticalthinking_2	Coursera	15,627	5
C_{14}	criticalthinking_3	Coursera	11,761	5

Figure 1: Summaries of courses used for experiments. The first set contain five courses from edX platform (Harvard-MIT), the second set contain ten courses from the EDI platform (University of Edinburgh).

We assume that the *source* courses and the *target* course share a non-empty set of behavioral features, such that we can restrict ourselves to this set when building our predictive models. As we will see below, this hypothesis is often verified in practice. In this context, our goal is to learn the statistical mapping from the behavior vector $x^{w'}$ of a student in week w' to their particular outcome y^w in week w. To do this, we propose to learn a statistical model by leveraging data (both $x^{w'}$ and y^w) from previous courses.

Data sets: Our experiments are based on two sets of MOOC courses. The source set, on which we built and validated our methods, consists of five courses, and was provided by the edX platform. Its attributes are described in table 1. This dataset initially contained log files describing students' behavior on the platform. For each student in these courses, we extracted a set of 21 features on a weekly basis. The second, or "target," set of courses was given by the University of Edinburgh (EDI) and consists of 10 courses from Coursera, whose attributes are also described in table 1. The courses in this second set are shorter in duration (only 5 weeks), and contain less detailed features. They share only 11 features with the first set of courses.

To build a robust framework that could achieve reliable predictive models, we initially designed, trained and validated our different methods on the first set of 5 courses from edX. At the very end of this paper, we apply these models to the second set of courses. When building models on one course and applying its predictions to another, two issues must be overcome. First, the two courses might not share the same features (for example, the grade for p-sets during week 1 might be available for some courses and not for others). Second, they might not span the same number of weeks. We overcame these issues by only considering features and timespans common to all courses. Therefore, we first used 21 features and 9 weeks when we trained, tested and validated our models on the edX courses. We then restricted ourselves to an 11-feature, 5-week scheme when testing our procedure on the EDI courses.

3. METHODS

In this section, we describe the different approaches used to build a predictive model for dropout. We first describe common practices, and explore whether a single course can be used to build a predictive model for another course. We then explain how the aggregation of several data sources can be used to improve the predictive power of a model. Finally, we describe how a type of machine learning technique called "Ensembling methods" can be used to further boost the predictive power of models built from different courses.

3.1 Naive approaches

Simple models: When building a supervised predictive model out of data sources, the first logical step involves training a single model on a particular dataset. Although plenty of classification algorithms exist, there is no systematic a-priori method to determine which one is best suited to a particular problem. This is the first hurdle that must be overcome when building a robust model.

The second hurdle involves choosing which prior course to train the model on. Although the first course s_1 may have a distribution closer to that of our target course t, s_2 may have more samples, resulting in a better predictive model. Hence, we must choose both an algorithm and a prior course that, working together, will be most suitable for predicting outcomes in the new course.

Merging sources: Alternatively, one may ask, why not use all the data from all the courses? Could learning a predictive model on the concatenated data from all courses $\{s_1, s_2, ..., sn\}$ result in a model that transfers better to new courses? This mitigates the problem of choosing among courses, but certainly does not solve the need to choose a a modeling approach, as it raises a number of new questions. First, concatenating obscures the differences between courses, preventing a predictive model from making predictions within the environment of the original sample. Second, if the courses have different numbers of students, concatenating them can overweight the influence of the larger data sets. Though this may not be a concern in cases where all datasets are drawn from a single distribution, in our case, combining the datasets is likely to limit the overall information available.

Although those concerns could be addressed using different tricks (for example, adding a "dataset" feature to account for the particularities of models, or undersampling bigger datasets to balance their weight in the concatenated set), we instead sought a different and more systematic approach to building robust models.

3.2 Ensembling methods to improve transfer of models in MOOCs

In this section, we leverage a type of machine learning technique called "Ensembling methods," often used to aggregate different predictive models. These techniques are now widely practiced after their successful deployment in the *Netflix*[1] challenge, in which hundreds of teams competed to build a precise recommendation system. They are used both in the industry and in public competitions, such as those held by Kaggle [2]), to improve the predictive power of models trained and tested on a single dataset([3]).

In this paper, we ask whether ensembling methods can in fact help in transferring models trained on one or more courses. What additional parameter tuning or methods do we have to develop to make this transference possible? Ordinarily, a data scientist uses ensembling techniques by selecting different subsets of features and training examples, learning algorithms, or parameters, and then building a set of predictors to ensemble. In the context of MOOCs, which have multiple courses, there is a natural split in the data we can exploit. We will demonstrate that in the some cases (for short term predictions), these approaches outperform the performance of the transferred predictive models built on a single course data and from a single algorithm.

We will discuss the different methods explored with respect to the three following dimensions :

- A set of pre-trained predictors $E = \{p_1, ..., p_n\}$
- A set of rules to combine the predictions of different algorithms. We call these rules "voting rules" and note them $R = \{R_1, ..., R_p\}$
- A structure S, which specifies in which order and to which predictions these rules should be applied.

Predictors: The first step in building a transferring method for dropout prediction is to train a set of predictive algorithms on data available from past courses. Given N source courses and P predictive models to train, this produces $N \text{x} P = H$ predictors $\{p_1, ..., p_H\}$.

We trained four classification algorithms (RandomForest, Logisitic Regression, SVM, and Nearest Neighbors) on each course. For each of these algorithms, we used 5-fold cross-validation to optimize the parameters. We note that for each of the past available courses, we left a holdout subset of 20% for a later-stage parameters optimization.

Fusing methods: One can combine a set of underlying predictions $\{p_1(x), ..., p_H(x)\}$ in infinite ways. Below, we list three common ways of ensembling that have been proven to perform well over a broad range of applications:

- Averaging (R_1). The most common fusion method consists of averaging the predictions of different underlying predictors.
$$p_{norm}(x) = \frac{1}{H} \sum_{i=1}^{H} p_i(x)$$
- Normalized averaging (R_2). When combining disparate predictive methods, some predictors might produce estimations in $\{0.49, 0.51\}$, while others produce estimations in $\{0, 1\}$. To account for the diversity of ranges from one predictor to the next, one can normalize the predictions of each predictor before averaging them.
$$p_{avg}(x) = \frac{1}{H} \sum_{i=1}^{H} \frac{p_i(x) - \min_{z \in t} p_i(z)}{\max_{z \in t} p_i(z) - \min_{z \in t} p_i(z)}$$
- Rank voting (R_3) In addition to differences in the range of probabilities, may also differ in how fast they

[1] http://www.netflixprize.com/

[2] https://www.kaggle.com/

[3] https://inclass.kaggle.com/c/mooc-dropout-prediction

vary with the input. To mitigate this behavior (which might cause the overall prediction to overweight very sensitive predictors), one can rank the probabilities within the target set first, then average and normalize the resulting ranks of different .

$$p_{rank}(x) = \frac{1}{H}\sum_{i=1}^{H}\frac{rank(p_i(x)) - 1}{N_t}$$

where $rank(p_i(x))$ refers to the rank of sample $p_i(x)$ in the set $\{p_i(z), z \in t\}$

We note that none of these techniques assume anything about the relative performance of different algorithms. We call those voting schemes "symmetric" because they treat each predictor in the same way. Our next set of methods allows us to fuse predictions by accounting for the varying performances of different predictors, and allowing us to put more weight on the "best" predictors. To identify these weights, we use the holdout subset of our source courses, and develop a method known as *stacking* as follows:

- Stacking (R_4) We concatenate all the holdout subsets from all source courses $X_{HO} \in R^{N_{HO} \times d}$ and apply all pretrained predictors $\{p_1(x), ..., p_H(x)\}$ on this dataset. The output of this procedure $Y_{HO} \in R^{N_{HO} \times H}$ is then considered as a new training dataset. We apply a logistic regression on this output to learn the weights for each predictor.

Structures: The last component of an ensembling method is the structure, within which predictions are merged together. Two example structures are shown in figure 2. Structures can influence the final performance of the method. Given a set of predictors and a set of fusing methods, the

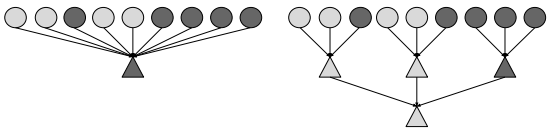


Figure 2: Illustration of two structures used to combine the same set of predictions using a simple voting rule (R_1) (color code is 1 for blacks and 0 for whites). The two different structures result in two different predicted outcomes.

"structure" is the sequence in which said predictors are fused in order to produce the final output.

Learning the structure: We posit that the structure of votes could influence the performance of the overall ensembling method. Due to the potentially arbitrary number of "layers," the number of possible structures is infinite. We restrict ourselves to structures with a high degree of symmetry. We enumerate a subset of structures in the figure 4. We then use algorithm 1 to compare the performance of the preselected structures. Our goal is to find the structure that will yield the highest performing predictor when applied to target courses.
For this comparison to be independent of the choice of target course, we consider each one of the five edX courses as the target course successively, calling them C_0, C_1, C_2, C_3 and C_4). The remaining four act as source courses. We then

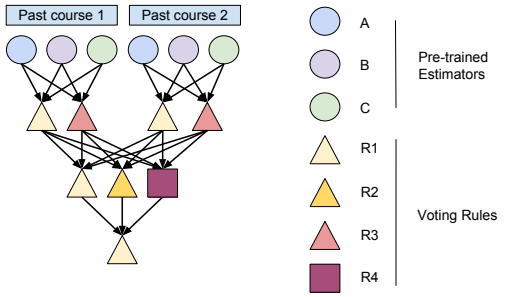


Figure 3: Example of a complex voting structure built on top of two data sources (past courses). The predictions of the different predictors are first aggregated by course then aggregated across courses.

aggregate our results by averaging the performance over the five permutations. We also remark that, in order to learn the metamodel necessary to the stacking rule, we separate all source course into a train and a validation set, as explained in algorithm 1.

Data: Full Data for the 5 edX courses
Result: Evaluate performance of structures
for *problem in P* **do**
  initialization : Split each dataset into a training and a validation subset (80% - 20%);
  for *t in set of courses* **do**
    Train each of the predictive algorithms (Random Forest, SVM, Logistic Regression, Nearest Neighbors) on each train set for the 4 remaining courses;
    for *s in set of Structures* **do**
      if *s requires training* **then**
        concatenate validation sets for 4 remaining courses;
        train s on this dataset;
      else
        pass
      end
      measure ROC AUC : $AUC_s^{p,t}$;
    end
  end
end
Algorithm 1: Comparing performance of different ensembling structures.

4. RESULTS ON DROPOUT PREDICTION

MOOC platforms offer courses that span a particular length of time, typically around 12 weeks. A large cohort of students register for each of these courses, but only a fraction of this cohort usually remains at the end of the class.
We consider the common problem of predicting which students will remain in the class. Specifically, given a "current week" w_c and a "prediction week" w_p our goal is to identify which of the students present in the class at week w_c will have dropped out by week w_p. We call this particular problem (w_c, w_p), and we remark that, given a particular course lasting W weeks, there exist exactly $\frac{W.(W-1)}{2}$ potential problems of this type.

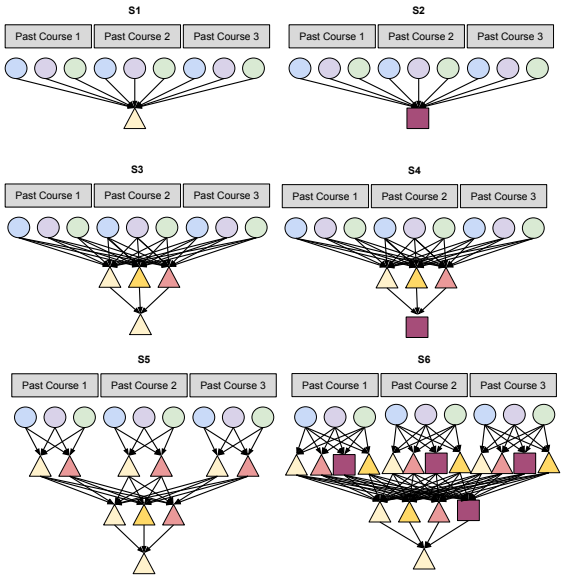


Figure 4: The six structures evaluated on the MOOCs dropout prediction problem. Only $S2, S4$ and $S6$ require training on the validation data because of the presence of a "stacking vote rule". (Note that for simplicity the diagrams shows only 3 courses and 3 predictive algorithms per courses whereas we used 4 and 4).

In the first to parts of this section, we use the five courses from edX (described in 1) to experiment and build our predictive models. We noted that the five courses from edX shared 21 behavioral features altogether. In the third part of this section, we show that these models indeed perform significantly better than our benchmark, even on courses from a different platform, Coursera.

4.1 Performance metrics and benchmarks

Evaluation metrics: To measure the performance of our predictive algorithm, we rely on the AUC-ROC metric, which is commonly used in dropout prediction. Because not all courses last for the same amount of time, we restrict ourselves to problems acceptable for all courses; i.e. the set

$$P = \{(w_c, w_p)\ s.t.\ w_c < w_p \text{ and } w_p < W_{course}\ \forall\ course\}$$

For the five courses used in this study, $W = 12$, meaning we can experiment on $|P| = 66$ different prediction problems. When comparing the performance of algorithms between problems, it becomes clear that some situations are intrinsically more difficult to predict than others. For instance, a short-term prediction problem (e.g, $(6, 5)$) will generally yield higher performance than a long term problem (e.g, $(6, 1)$). Similarly, some courses are more suited for predictions than others, due to the size of the student cohort or the volatility of students within that cohort.
To mitigate this, we *normalize* the performance, and use the following metric to measure the performance of an algorithm a on a problem p and on *target* course t :

$$DAUC_a^{p,t} = \max_{a' \in A} \left(AUC_{a'}^{p,t}\right) - AUC_a^{p,t}$$

In other words, we subtract the actual AUC of an algorithm from the best observed AUC of any other algorithm on this particular problem for this particular *target* course. In this configuration, a lower DAUC should be considered to indicate a better performance. In particular, $DAUC_a^{p,t} = 0$ exactly means that a is the best algorithm for this particular problem and target course.
To appreciate this metric over different problems, we display both the mean and the variance of the DAUC. In order to account for the different performance on different types of problems, we introduce two sets of problems, for which we choose to average the DAUC:

Two subset of problems

$\boldsymbol{P}$ Mean ROC AUC obtained on the 66 available problems

$\boldsymbol{P}_s$ Mean ROC AUC obtained on three 'short term' prediction problems ($\{(5, 6), (8, 9), (11, 12)\}$)

Benchmarks : A simple approach to building predictive models is to train a classifier on a *source* course and use it to make predictions on the *target* course. In figure 5, we report the results obtained by training four different classification algorithms on a *source* course (for course 1 to 4) and applying it to the *target* course (C_0). We use 5-fold cross-validation on the training set, and we tune the parameters independently for each method, each source and each prediction problems.
A more systematic approach consists of building predictive models on the concatenation of all available data. In addition to avoiding the hurdle of having to 'guess' which course should be chosen as the *source* course, this approach also allows us to leverage more (and more diverse) data to train predictive models.
As shown in figure 5, we observed that, regardless of the algorithms used, models trained on the concatenation of all available data sources always performed better than the best models trained an a single course. This is true both in terms of average DAUC over all problems in P, and in terms of variance of the DAUC across those same problems.

4.2 Building robust models

Improvement through Merging Methods :
Concatenating the data from past courses undoubtedly improved the algorithms' predictive power, both in terms of average DAUC and variance. To further improve the average performance, and to reduce the variance of our dropout prediction system, we then leveraged the ensembling methods presented above. Instead of restricting ourselves to a choice of a single predictive algorithm, we trained four of them (SVM, Random Forest, Logistic Regression and Nearest Neighbors) and merged their predictions using a simple R_1 voting rule.
Figure 5 shows the average DAUC and its variance for different algorithms, as well as their "merged" version through an R_1 rule. Comparing the result obtained by the "merged" method with those of the four single algorithms, we observe that the merged method always performs comparably to the best single algorithm, beating all competitors on courses C_2 and C_4, and behaving comparably on courses C_1 and C_3). This is true both in terms of average performance and in terms of variance.

Next, we apply this same R_1 rule to merge the predictions built on the data concatenated from all available *source* courses. Here, the results unveil a lower DAUC average and variance for the "merged" method on the concatenated data than for any other algorithm on the same data. Moreover, this method performs better than those "merged" methods trained only on a single course, in terms of both average and variance. Through an "all-algo all-data" kind of method, we have achieved a more reliable and more accurate predictive model, on average, over all possible prediction problems.
In the next section we will see that, for certain type of problems, it is possible to improve this model significantly by using a more complex type of ensembling method called stacking.

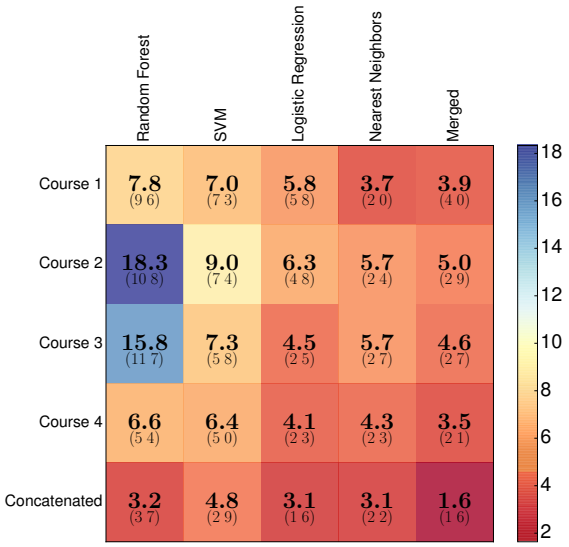


Figure 5: Average (standard deviation) of the DAUC (x100) for all predictions problems (P) on target course C_0. The x-axis contains different predictive algorithms, the y-axis contains different data source.

Optimizing the vote Structure:
The figures above show promising results for ensembling methods in the context of dropout predictions. This encouraged us to explore different ensembling methods to further improve the and performance and/or reliability of our dropout prediction system.
Our ensembling strategy uses all available estimators described above (those built on a single-source course as well as those built on the concatenated data). It then applies one of the manually pre-selected structures as shown in figure 4. We then use algorithm 1 to learn the structure.
Figure 6 displays the DAUC obtained by different ensembling structures according to algorithm 1. We differentiate our observations according to the subset of problems over which the average is computed (P,P_s and P_l).

- Over all problems (average over P), we first remark that the structure has only a very small impact on both the average performance and the variance of the predictive method. We also remark, however, that structure $S6$ yields slightly better results, both in terms of average DAUC and in terms of variance.

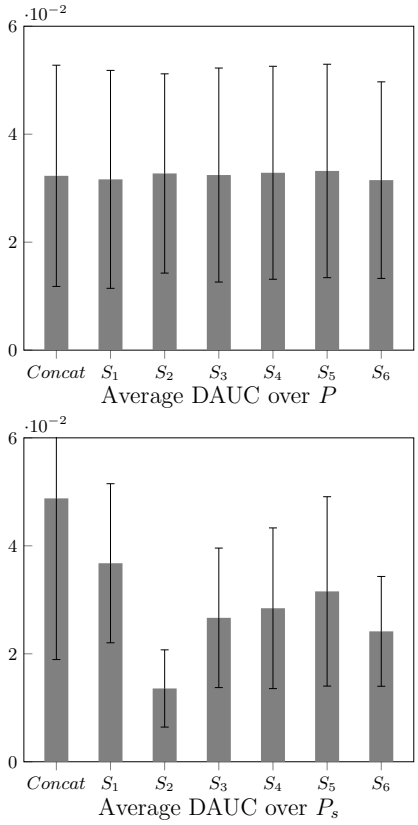


Figure 6: Average DAUC over each edX course taken as target, as computed by algorithm 1.

- Over the short-term problems, (average over P_s), we observe a lot more difference across the different structures. By far, structure $S2$ is the best performer for this type of problem, with a DAUC of 0.013 on average compared to 0.049 for the merged method discussed in the previous section.
- Over long-term problems (average over P_l) the difference between structures is significant, and thus not as big as for the short term problems. The best structure here is $S4$.

4.3 Transferring across MOOC platforms

Having achieved robust methods for dropout prediction on different predictive problems, we now test our method on a new set of courses, composed of 10 courses from the University of Edinburgh. Rather than testing this method on the holdout course as explained in algorithm 1, this set of courses present the additional difficulty of being derived from another MOOC platform (Coursera), thus having potentially very different statistics for the features used in our models. For example, overlap between the features of our 5 first courses (from edX) and the features of those new courses is not total. Whereas our initial 5 courses shared 21 common features, they share only 12 features with this new set of courses. In figure 7 we report the DAUC obtained on average over all possible prediction problems and over all ten

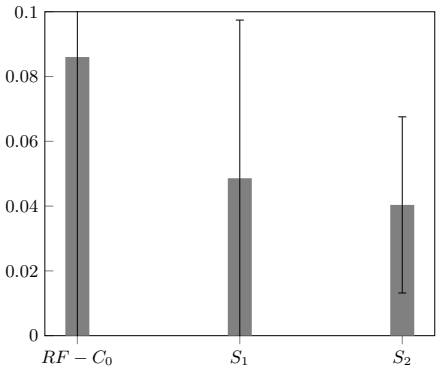


Figure 7: Average DAUC over all prediction problems on the 10 UDI courses taken as target (only edX courses are taken as source courses).

ID	Name	AUC short	AUC all
C_5	aiplan_001	0.82	0.75
C_6	aiplan_002	0.79	0.70
C_7	aiplan_003	0.81	0.74
C_8	animal_001	0.73	0.64
C_9	animal_002	0.75	0.67
C_{10}	astrotech_001	0.77	0.67
C_{11}	codeyourself_001	0.84	0.74
C_{12}	criticalthinking_1	0.71	0.63
C_{13}	criticalthinking_2	0.80	0.71
C_{14}	criticalthinking_3	0.78	0.70

Figure 8: AUC achieved by S_2 ensembling method built on the five edX courses and applied on the 10 UDI courses.

courses (taken as target). We display the results for a simple Random Forest algorithm built on the first edX course, for an ensembling method based on the S_1 structure, and finally for the best-performing ensembling method (from the experiment in the previous sub section) based on S_2 structure. All the ensembling methods are built on top of estimators from all the five edX courses (for the four algorithms : Rrandom Forest, SVM, Logistic Regression, Nearest Neighbor). Table 8 reports the absolute performance of the best technique (S_2) structure in terms of average AUC across different prediction problems for each course.

We remark first that the S_2 performs again significantly better than both the simple algorithm and the simple ensembling method. We also note that the absolute performance achieved by this best ensembling technique is relatively high, given the small number of features available and the different origins of the two set of courses.

5. KEY FINDINGS

Our key findings can be summarized in three categories, corresponding to the three sets of questions described in the introduction:

Purpose We showed that even though MOOC courses span different numbers of weeks and have different characteristics, one can usually find sufficient overlap between courses to perform nontrivial prediction tasks.

Data We showed that using more courses as training data improved the predictive power significantly. We also proved that this predictive power was sufficient to apply the model built on one particular MOOC platform to another platform.

Method First we showed that, both in the case of a single course model and in the case of a model built from several courses, using simple ensembling methods between algorithms significantly improved the performance. When compared to a single algorithm trained on all available courses, a simple ensemble methods improved the AUC by an average of 1.5 to 3.2 points. Secondly, we proved that in certain use cases (for instance, short term dropout prediction problems), using more complex ensembling structure can significantly boost performance. For short term prediction problems, using a S_2-like structure of ensembling resulted in no less than a 4 point AUC improvement on average.

Finally, our best method was successful when applied to a set of unseen courses. On the ten never-before-seen Coursera courses, our method obtained a 0.70 average AUC overall and a 0.78 average AUC on short term prediction problems (one week ahead). This completes our case that a high-performance predictive model can be built from a set of previous courses, and that ensembling methods appear to be a suitable framework to build such models.

6. DISCUSSION

When trying to estimate the actual benefit of such techniques on the real life of students and teachers on MOOC platforms, one has to make several assumptions that may only be verified after several years of implementation.

The main assumption is the possibility to reduce churn of

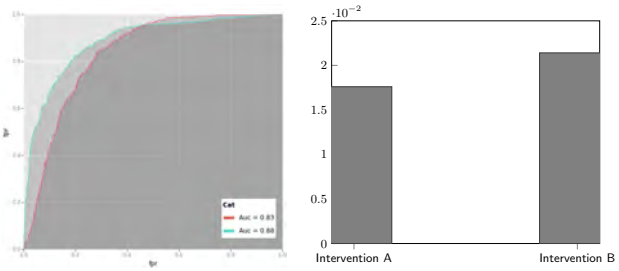


Figure 9: Estimated increase in number of students completing a typical MOOC course. Intervention A refers to an intervention based on a simple predictive model built on a course. Intervention B refers to an intervention of type S_2. See description for numerical assumptions.

students through personalized intervention. This is not obvious, as many argue that most dropout students were intrinsically not interested in the content of the class, and could therefore not be fruitfully intervened with. Most MOOC providers, however, agree that a good chunk of each cohort could be prevented from dropping out of the class thanks to some customized and well-adapted interventions. Identifying dropout students (the example describe in this paper) could enable a concrete set of interventions to be done, with extra resource help, additional videos or motivating resources particularly tailored to potential "dropout" students. For our purpose we will assume that a tailored intervention will save 1% of all potential dropout students.

Given a fixed false-positive rate, arguably necessary to design an intervention targeted for dropout students, the purpose of the predictive methods described above can be understood as the maximization of the true-positive rate: the ratio of predicted dropout students to the number of total dropout students.

Taking a weekly intervention framework, in which an intervention is conducted for potential dropout students at the end of each week, we showed in the previous section that ensembling methods (particularly the S_2 structure) were able to perform around 0.05 AUC point better than other more straightforward models (particularly an "all-algo all-sources" method). In figure 9, we show the example of two ROC AUC separated by 0.05. We remark that with a constraint of 10% on the false positive rate, we obtain a difference of around 20% in the true positive rate .

Given a typical MOOC class − 10 weeks long, starting with 10.000 students, and with a typical weekly dropout rate of 20% per week we display in figure 9 the simulated data of the number of students completing the course. When an intervention based on a straightforward predictive model is simulated, it increases the number of students finishing the course by around 1.7%, whereas an S_2 based predictive model would increase it by around 2.1% (an additional 50 student completions overall).

7. CONCLUSION

In this paper, we developed a framework to address the main challenges faced when applying predictive analytics to MOOCs: How to build models that transfer well across courses and platforms?

To do this, we used ensembling methods, as well as a broad range of algorithms and a rigorous training procedure. We explored different variations of these techniques and reported the results obtained on a first set of five courses from the edX platform. We introduced a novel performance metric, allowing for performance comparison across prediction problems and target courses. These results show that ensembling methods improved the accuracy of prediction, both on average and in terms of variance. We also showed that "stacking" (or learning metamodels on top of a set of base predictors) can significantly boost performance in the case of short term prediction problems.

Eventually, we tested the method developed in a first part (on the first set of five courses from edX MOOC platform) on ten courses from the University of Endinburgh MOOC platform. We reported the results obtained in terms of AUC and showed that the method developed performed very well on those new courses, too.

We argue that our paper demonstrates a robust framework to develop predictive algorithms that are transferable across online courses.

8. REFERENCES

[1] P. Adamopoulos. What makes a great mooc? an interdisciplinary analysis of student retention in online courses. 2013.

[2] S. Boyer, B. U. Gelman, B. Schreck, and K. Veeramachaneni. Data science foundry for moocs. In *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*, pages 1–10. IEEE, 2015.

[3] S. Boyer and K. Veeramachaneni. Transfer learning for predictive models in massive open online courses. In *Artificial Intelligence in Education*, pages 54–63. Springer, 2015.

[4] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes. Ensemble selection from libraries of models. In *Proceedings of the twenty-first international conference on Machine learning*, page 18. ACM, 2004.

[5] S. Džeroski and B. Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine learning*, 54(3):255–273, 2004.

[6] S. Halawa, D. Greene, and J. Mitchell. Dropout prediction in moocs using learner activity features. *Experiences and best practices in and around MOOCs*, 7, 2014.

[7] I. Lykourentzou, I. Giannoukos, V. Nikolopoulos, G. Mpardis, and V. Loumos. Dropout prediction in e-learning courses through the combination of machine learning techniques. *Computers & Education*, 53(3):950–965, 2009.

[8] S. J. Pan and Q. Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[9] B. Poellhuber, M. Chomienne, and T. Karsenti. The effect of peer collaboration and collaborative learning on self-efficacy and persistence in a learner-paced continuous intake model. *International Journal of E-Learning & Distance Education*, 22(3):41–62, 2008.

[10] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos, and P. Stamatopoulos. Stacking classifiers for anti-spam filtering of e-mail. *arXiv preprint cs/0106040*, 2001.

A Comparative Analysis of Techniques for Predicting Student Performance

Hana Bydžovská
CSU and KD Lab Faculty of Informatics
Masaryk University, Brno
bydzovska@fi.muni.cz

ABSTRACT

The problem of student final grade prediction in a particular course has recently been addressed using data mining techniques. In this paper, we present two different approaches solving this task. Both approaches are validated on 138 courses which were offered to students of the Faculty of Informatics of Masaryk University between the years of 2010 and 2013. The first approach is based on classification and regression algorithms that search for patterns in study-related data and also data about students' social behavior. We prove that students' social behavior characteristics improve prediction for a quarter of courses. The second approach is based on collaborative filtering techniques. We predict the final grades based on previous achievements of similar students. The results show that both approaches reached similar average results and can be beneficially utilized for student final grade prediction. The first approach reaches significantly better results for courses with a small number of students. In contrary, the second approach achieves significantly better results for mathematical courses. We also identified groups of courses for which we are not able to predict the grades reliably. Finally, we are able to correctly identify half of all failures (that constitute less than a quarter of all grades) and predict the final grades only with the error of one degree in the grade scale.

Keywords

Student performance prediction, student similarity, classification, regression, collaborative filtering.

1. INTRODUCTION

One of the key problems of educational data mining is to design student models that would predict the student performance. Once we have a reliable performance prediction, it can be used in many contexts: for identifying weak students [14], for guiding the adaptive behavior in intelligent tutoring systems [10], or for providing a feedback to students.

Our specific problem is the following: we have access to data about students, their study achievements and their behavior characteristics stored in the university information system and we want to predict students' final grades. The predictions are useful at the beginning of each semester to help students with planning their workload in the whole semester. We also beneficially use this information to design a course enrollment recommender system. The early grade prediction is more difficult since we have no a priori information about students' knowledge, skills or enthusiasm for particular courses. It has been proven [4] that the data about the activity of students during the semester improves the prediction.

The problem of the student grade prediction in a particular course has recently been addressed using data mining techniques. Researchers usually examine study-related records, e.g. the age, the gender, and the field of study [9] because of their easy availability in university information systems. Moreover, they attempt to identify additional characteristics that can lead to better understanding of students' behavior, e.g. their habits [6] or parents' education [13]. The most typical way how to obtain such data is to conduct questionnaires. Masaryk University has more than 40,000 active students and we try to predict the grades as accurately as possible for all of them. We cannot rely on data obtained by questionnaires since they tend to have a lower response rate. Therefore, only the data originated from the Information System of Masaryk University (IS MU) are employed for our experiments.

The goal of this research is to predict students' grades with the major emphasis on the detection of students who can fail to meet the course requirements. Therefore, we are dealing with the following two main tasks:

- prediction of students' success or failure,
- prediction of the students' final grades.

In this paper, we present two different approaches moving towards our objectives. The first approach is based on the state of the art educational data mining techniques: classification and regression analysis [12]. We created an ensemble learner to utilize the strength of the both techniques. We also present a new type of data about students' social behavior originated from IS MU that can improve the predictions. The second approach is based on collaborative filtering techniques [5] applied to the educational context. We mapped the users-item-rating problem to the student-course-grade problem and predict the final grades based on previous achievements of similar students. This paper describes both approaches in detail, compares them and reports their advantages and disadvantages.

2. DESIGNED METHODS EVALUATION

Historical data were employed for experiments allowing us to evaluate both designed approaches. We processed data about 138 courses which were offered to the students at the Faculty of Informatics. We used only data stored in IS MU in the time of students' enrollments. We omitted freshmen students because we had no data about them in the system. The data comprised of 3,584 students. The two independent data sets were used. The training set consisted of the data collected between the years of 2010 and 2012 (37,005 instances) and was used for the identification of the most suitable methods with their settings. The test set consisted of the data from the year 2013 (11,026 instances) and was used for the validation of the methods on different data.

The following grade scale was used: 1 (excellent), 1.5 (very good), 2 (good), 2.5 (satisfactory), 3 (sufficient), 4 (failed or waived). The value 4 represents student's failure; the others represent a full completion. We evaluated approaches using the *mean absolute error* (MAE). The technique measures how close predictions are to the real outcomes. Lower values represent better

results. The measure is commonly used for grade prediction evaluation. In the educational environment, one of the most important issues is to reveal weak students. Therefore, we also computed the *sensitivity* (also called recall). Categorizing students only as successful or unsuccessful, the sensitivity measures the proportion of unsuccessful students who are correctly classified as unsuccessful. For students' success or failure prediction we also utilized *F1 score* that conveys the balance between the precision and the recall.

3. STUDENTS' CHARACTERISTICS

3.1 Study-related Data

Classification and regression are the most often used techniques for student performance prediction [12]. Researchers usually examined study-related (SR) data. Our study-related data contained common attributes such as the gender, the year of birth, the year of admission, the number of credits gained from passed courses, or the average grades. We built a classifier for each investigated course based on the training set and evaluated the results using the 10-fold cross validation. The method that achieved best results was subsequently validated on the test set.

3.1.1 Student success/failure prediction

The first task was to reveal unsuccessful students. Two prediction classes were considered: students success def grades 3) and failure (def. 2: grade 4). Widely utilized classification algorithms were employed: Support Vector Machines (SVM), Random Forests, Rule-based classifier (OneR), Trees (J48), Part, IB1, and Naive Bayes (NB). As the baseline we defined a model which always predicts failure. Table 1 confirms that SVM achieved the best performance.

Table 1. Classification algorithms results

Rank	Method	F1	MAE	Sensitivity
1	SVM	0.559	0.161	0.444
2	NB	0.554	0.251	0.467
3	J48	0.552	0.182	0.397
4	Random Forests	0.550	0.173	0.362
5	Part	0.543	0.202	0.417
6	IB1	0.536	0.216	0.436
7	OneR	0.508	0.183	0.321
8	Baseline	0.326	0.822	1

3.1.2 Grade prediction

The regression is a commonly used technique for student grade prediction. Widely utilized regression algorithms were selected: SVM Reg., Random Forest, IBk, RepTree, Linear Regression, and Additive Regression. The baseline model predicts the average grade of the training set of a given course. The best results (see Table 2) were achieved by support vector machine (SVM Reg.).

3.1.3 Conclusion

For each task, the best method was selected and an ensemble learner was built. If the classifiers (SVM or SVM Reg.) predicted the failure or the grade 4, then the ensemble learner also predicted the failure. Otherwise, it resulted in the value of the grade predicted by the SVM Reg. classifier. Finally, the overall performance of this approach could be seen in Table 3. We also evaluated the classifiers on the test set. The results indicated that we were able to reveal almost half of the unsuccessful students even if the task was difficult due to the fact that all unsuccessful students constitute less than a quarter of all students. The prediction error was about 0.75 on average which was almost 1.5 degree in the grade scale.

Table 2. Regression algorithms results

Rank	Method	MAE	Sensitivity
1	SVM Reg.	0.605	0.196
2	Linear Reg.	0.615	0.152
3	Additive Reg.	0.634	0.165
4	RepTree	0.643	0.184
5	Random Forests	0.668	0.216
6	IBk	0.767	0.294
7	Baseline	0.806	0

Table 3. Global SVM results

Data Set	MAE	Sensitivity
Training Set	0.701	0.524
Test Set	0.744	0.414

3.2 Social Behavior Data

Recent researches are often based on finding additional data that can improve the prediction accuracy. Our improvements have been achieved through adding social behavior (SB) data to the original data set [1]. This specific type of data originating from IS MU described the students' behavior characteristics and their mutual cooperation. We focused on statistical data that represented an interaction among students: posts and comments in discussion forums, e-mails statistics, publication co-authoring, or files sharing. This information served as the basis for computing social ties among students and building a sociogram. From this sociogram, new features like weighted average grades of friends can be easily derived. Using Pajek [11], we also computed additional standard graph features [3] like degree (the number of the friends), weighted degree (degree weighted by the strength of ties), centrality or betweenness (the importance measure for each student in the network). Moreover, we collected data about students' disclosure from different system sections. By default, IS MU does not provide a complete list of classmates due to the students' privacy. Students have to actively disclose themselves to become visible for their classmates. We can also calculate how many times students attended courses of a certain teacher. Among others, students can also mark offered courses as favorite.

H Hypothesis supposes that students' social ties correlated with the students performance.

Other ensemble learners trained on data sets containing social attributes were built. The other settings were maintained. The comparison of the results can be seen in Table 4. The MAE score was slightly lower on average. However, for 32 courses in the test set, the difference in MAE was significantly better using social behavior data (min: 0.1; average: 0.178; max: 0.734). Only 5 courses achieved worse results (min. 0.1; average: 0.12; max: 0.21). For the rest courses, the difference was negligible.

Table 4. Adding social behavior attributes to the data set

Data Set	Attributes	MAE	Sensitivity
Training Set	SR	0.701	0.524
	SR + SB	0.629	0.528
Test Set	SR	0.744	0.414
	SR + SB	0.688	0.427

The sorted list of selected attributes was constructed. In Table 5, we present the top five social behavior attributes that significantly affected the results.

Table 5. The most interesting social behavior attributes

Rank	Avg. Ord.	Attribute
1	13.328	the betweenness
2	16.252	the information if the course was marked as favorite
3	18.694	the centrality
4	22.464	the weighted degree
5	29.807	the number of times when a student attended any course with the same teacher

H1 was confirmed. Data about students' behavior improved the predictions. Based on the most significant attributes, we assumed that the assistance of students' friends had increased the probability to pass the courses.

4. STUDENTS' GRADES

We also focused on methods utilized in recommender systems [5]. The data about user-item-rating triples were replaced by student-course-grade triples and we focused on the similarities among students' grades.

H2: Our hypothesis supposed that students' knowledge can be characterized by the grades of courses that students enrolled during their studies. Based on this information we could select students with similar interests and knowledge and subsequently predict whether a particular student has sufficient skills needed for a particular course.

4.1 Grade Prediction

Our preliminary work can be found in [2]. However, the approach suffered from several limitations that we overcome in this paper.

The first step was to build a similarity matrix *G* where rows represented students and columns represented courses. Although we predicted grades for 138 courses, the matrix *G* has 499 columns since we analyzed all students' grades (e.g. courses from the other faculties, courses not offered now). Grades obtained by all students from the training set formed the matrix. If a student did not attend a particular course, the corresponding cell remained empty. The aim was to complete cells defining students' grades from the investigated courses enrolled by students in 2012 (marked by symbol *?*).

Using the vectors of grades from the matrix *G*, we computed the similarity between all students enrolled in a course *c* in 2012 and all students previously also enrolled in *c* in 2010 or 2011.

Example of Matrix G

Students / Courses	c_1	c_2	c_3	c_4
s_1	2	?		?
s_2	?	2.5	3	?
s_3	1		2.5	3
s_4		2		1.5

Widely utilized similarity metrics were used for the calculation of the students' similarity: Mean absolute difference (MAD), Root mean squared difference (RMSD), Cosine similarity (COS), and Pearson's correlation coefficient PC) All metrics compare grades of students' shared courses The average number of courses shared by students was 10.

Subsequently, the appropriate neighborhood of the most similar students to the examined student could be selected to influence the predicted final grade. We utilize the idea of a baseline user [7]. We selected such students to the neighborhood who were more similar to the investigated student than the investigated student was to the baseline student. We decided to calculate two types of baseline students: an average student (the average grade for each course) and a uniform student (the average grade through all courses: 2.5). The neighborhood of the top 25 students showed reasonable results. However, for smaller courses, 25 students could be all students enrolled in the course in one year. Therefore, we have decided to define three categories of courses with respect to the course occupancy: small (≤ students), medium 70 students), and large ≥70 students). Therefore, we analyzed the suitable size of the neighborhood for courses with the different occupancy. Figure 1 shows the relationship between MAE and the cardinality of *N*. We selected the size of neighborhood as follows: 10 for small courses, 15 for medium courses, and 30 for large courses. In the figure, we can also see that the prediction for smaller courses was the most challenging.

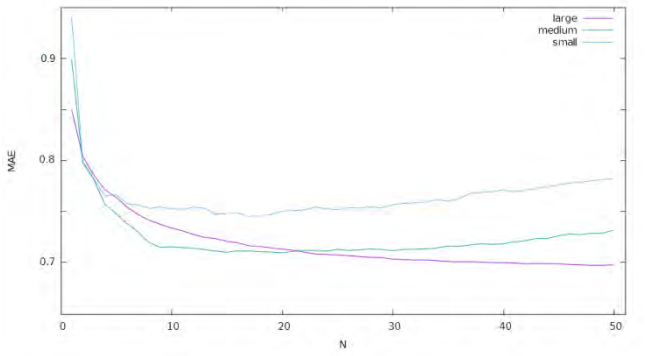


Figure 1. Relationship between MAE and the size of neighborhood with respect to the course occupancy

The final grades were estimated from the grades of similar students belonging to the computed neighborhood. Simple methods as mean, max, median as well as advanced methods utilizing significance weighting were utilized.

Table 6 introduces the top five combinations of the similarity methods, methods for the neighborhood selection and the grade estimation functions. The method utilizing a baseline user needed a large neighborhood for each student ($|N|$ = 376 on average). In the production system, it was very important to lower the ties

among students due to the recalculation of all similarities in the system during the course enrollment process to be up to date for students. Therefore, different neighborhood was selected even if the MAE score could be slightly higher. For efficiency reasons, we selected the third one for the implementation in the system.

Table 6. Similarity methods comparison

Rank	Method	|N|	MAE	Sensitivity
1	PC + average student + sig. weighting	376	0.648	0.248
2	PC + uniform student + sig. weighting	378	0.648	0.248
3	PC + Top |N| + sig. weighting	10/15/30	0.650	0.267
4	RMSD + Top |N| + median	10/15/30	0.651	0.211
5	PC + Top 25 + Pred	25	0.657	0.274

4.2 Student Success/Failure Prediction

The majority of students passed examined courses. Therefore, we searched for a smaller neighborhood in order to reveal more unsuccessful students. As you can see in Figure 2, the highest F1 was reached when we included only the most similar student. However, the method suffered by a low precision. Therefore, we predicted failure even if the method for grade prediction (3rd row Table 6) predicted grade worse than 2.4 (average grade). The precision was improved and still we found the sufficient number of unsuccessful students. The final results of methods were: MAE = 0.174, sensitivity = 0.413.

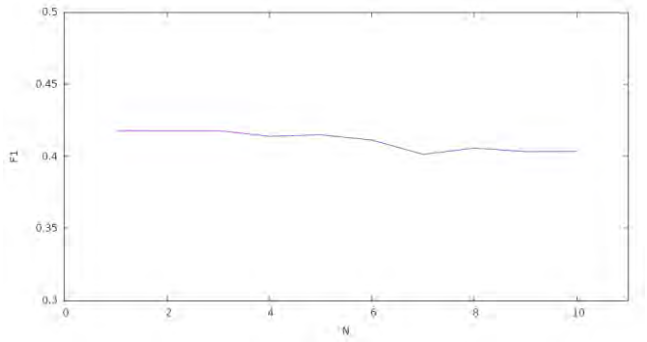


Figure 2. Relationship between F1 and the size of the neighborhood

4.3 Course similarity

Any change in the similarity matrix G could lead to the recalculation since the similarity of students was calculated from all students' grades.

H3: Our third hypothesis supposed that similar courses required similar skills of students to pass. It should decrease the computational cost and do not significantly lower the prediction accuracy when we use only grades of similar courses for predictions instead of all attended courses.

4.3.1 Students' grades

The collaborative filtering approach based on similarity of item to item was utilized and the *adjusted cosine similarity* was computed from the previously defined similarity matrix G for each pair of courses. Subsequently, we utilized the average link clustering [8] to group the investigated courses based on this similarity measure. The resulted clusters defined the groups of similar courses.

Finally, when we predicted the students' grades of a certain course, we reduced the computations to the grades obtained from courses belonging to the same cluster as the investigated course. 110 of all investigated courses belonged to one of the 37 clusters. The number of courses in one cluster ranged from 2 to 15. The average number of courses in one cluster was 3. The average number of students' shared courses was also 3.

4.3.2 Course Characteristics

Students search for useful information about courses in the Course Catalog that help them to decide whether or not they should enroll the course. We selected different course characteristics and attempted to identify dependencies among courses. Similarity of courses a and b was defined by the weighted sum of the similarities of the selected course characteristics $t \in T$:

$$sim(a,b) = \sum_{t \in T} w_t \ \mathrm{dist}(a_t, b_t)$$

where w defined the weight of the examined characteristic. The weights of the characteristics were set with respect to maximize the grade prediction accuracy. The similarity for each pair of courses was calculated. The selected characteristics and distance metrics *dist* were the following:

Prerequisites define a set of courses that had to be passed before students could enroll a certain course. The similarity was set to the value of 1 if the compared course belonged to the prerequisites; 0 otherwise. The weight of this characteristic was set to 1 because the prerequisites denoted a significant dependence.

Literature contains the recommended literature for particular courses that can be characterized by the set of assigned authors. The similarity of the set of authors A and the set of authors B is given by Jaccard's coefficient. The characteristics weight was set to the value of 0.9 due to the hypothesis that authors do not frequently publish in different fields. Therefore, the literature could constitute strong ties among courses.

The *course content* was represented by the text about the study subject and outline what students should learn in the course. We cut the STOP words from the text and utilized stemming to get the roots of the words. TF-IDF was utilized for defining the importance of each word in the texts. Subsequently, the Cosine similarity measure was used for the processing of the final vector representation of the words' importance. The characteristics weight was set to the value of 0.7.

Teachers of a course could be divided into two groups: lecturers and tutors. Weighted Jaccard's coefficient was used for comparing the teachers of the two courses. The weight of the lecturers was set to the value of 1 and 0.5 for seminar tutors. The weight of characteristic was set to the value of 0.6.

Course supervisor patronize the courses. The similarity was set to the value of 1 if the compared courses had the same supervisor; 0 otherwise. The characteristics weight was set to the value of 0.4.

When we calculated the similarity of courses by the aforementioned procedure, we could also utilize average link clustering [8]. 340 from all courses (499) belong to one of the 105 created clusters. 93 investigated courses were presented in one of the clusters. The number of courses in one cluster ranged from 2 to 22. The average number of courses in a cluster was 3. The average number of shared courses taken by students was 2.

4.3.3 *Comparison of approaches*

In comparison with the method using *all grades*, both approaches had positive effects on the number of calculations. 123 courses (from all 138) belonged to some of the created clusters and the final grades could be predicted based on the grades of only 3 other courses on average. 70 of our investigated courses belonged to different clusters using SC_1 and SC_2. A slightly better MAE was obtained by the method utilizing the course characteristics for these courses (see Table 7). Therefore, when a grade is predicted, the corresponding course is searched in SC_2, then SC_1.

Table 7. Comparison of SC_1 and SC_2

Method	MAE	Sensitivity	Average cluster size	Shared Courses
All grades	0.687	0.402	499	10
SC_1	0.681	0.390	3	3
SC_2	0.640	0.386	3	2

4.4 Conclusion

H2 and H3 were confirmed. We described the novel approach for predicting the students performance (see Table 8) using only students' grades and course characteristics. It proved to be as successful as the first described approach (see Table 9). The most important contribution of this approach was that each university information system stores the data about students' grades which were needed for the prediction unlike the data about students' social behavior. We also identified course dependencies that lowered the calculation cost. Moreover, we were able to predict the final grade considering grades from only 3 other courses for the most of the investigated courses.

Table 8. Global results of the approach

Data Set	MAE	Sensitivity
Training Set	0.661	0.470
Test Set	0.685	0.418

5. USAGE OF THE APPROACHES

Both approaches defined in Section 3 (based on students' attributes (SBA)) and Section 4 (based on students' grades (SBG)) reached similar average results (see Table 9). However, they can differ in specific situations. Our goal was to identify course groups for which we could get trustworthy predictions and also to detect when one approach outperforms the other.

Table 9. Comparison of the both approaches

Data Set	Approach	MAE	Sensitivity
Training Set	SBA	0.629	0.528
	SBG	0.661	0.470
Test Set	SBA	0.688	0.427
	SBG	0.685	0.418

H4. Each approach is more suitable for different course groups.

We selected the following categories based on the basic course characteristics:

- difficulty – the average grade of all students' grades is 2.4. Therefore, we divided courses into two categories: easy (≤2.4), and difficult (>2.4),
- occupancy rate – as defined in Section 4.1: small (≤), medium 7), and large ≥70),
- specialization – courses divided into four groups: mathematics (M), theoretic informatics (I), applied informatics (P), and others (O).

Each investigated course belonged to one of the groups for each of the defined categories. With respect to the three aforementioned categories, we could define six (3!) tree structures which differ in the splitting order of the categories. We examined each permutation of the categories. We built full trees where courses from the training set were split subsequently by all categories. Each node stored the information about courses that belonged to it with respect to the split. Harmonic mean (HM) was calculated for each node and both approaches in order to get a suitable relationship between the sensitivity and the MAE score.

Subsequently, we examined the trees and merged branches which were not interesting in order to detect significant phenomena. Interesting branches contained at least one of the following situations:

- Difference > 0.1 in HM of SBA and SBG in the node (The rule detected a significant difference in the prediction accuracy of the both approaches for the examined groups of courses.).
- Difference > 0.1 in HM of the sibling nodes (The rule detected course groups that were significantly easily or with difficulties predicted than other courses from this split.).
- Difference > 0.1 in HM of parent and child nodes (The rule detected the course groups that should be separated due to the significant difference in the prediction in comparison with the rest courses from the parent node.).

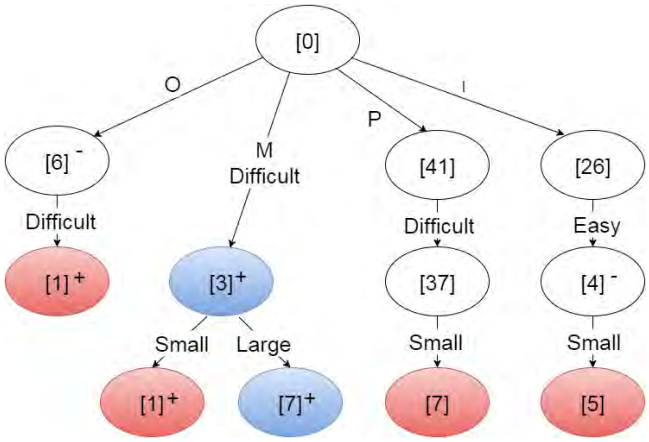


Figure 3. Resulted Tree

One of the resulted trees can be seen in Figure 3. As the figure shows, this approach had several benefits:

- Course groups that were predicted significantly better than average were identified (marked by +). It contains all mathematical courses (the main skill at the faculty of informatics can be easily predicted) and the English course.
- Course groups that were predicted significantly worse than average were identified (marked by). It contained almost all courses belonged to the category *others* (we do not know students' general knowledge) and medium or large easy theoretic informatics courses (the grade maybe depended on the amount of the effort which could differ for each course and cannot be predicted).

- H4 was confirmed. Course groups that were predicted significantly better by the SBG approach are represented by the blue color. It covered almost all mathematics courses (except one small course). Otherwise, red nodes present better results obtained by the SGA approach. It contained the most of small courses. For the white nodes, the difference in prediction accuracy was negligible.
- Outliers were also identified. One course of the group showed different behavior than others: the course of English (path: O-difficult) was easily predictable in comparison with all courses belonged to the category *others*; one small mathematical course (M-difficult-small) differed in the approach that achieved better results in comparison with all other mathematical courses.

We applied this knowledge for prediction of the students' performance when the test set was utilized. We can easily locate any particular course in the tree and used the suitable approach that led to the better results. We also gave no predictions for courses that we were not able to predict reliably. As the results in Table 10 show, MAE was significantly improved in comparison with the state of the art method utilizing only SVM. Finally, we were able to predict the final grades with an error of one degree in the grade scale. We were also able to reveal almost a half of the unsuccessful students.

Table 10. Final results validated on the Test set

Approach	MAE	Sensitivity	Omitted Courses
Novel	0.609	0.436	10
SVM	0.744	0.414	0

6. CONCLUSION

In this paper, we focused on the problem of predicting final grades of students at the beginning of the semester with the emphasis on identifying unsuccessful students. Two different approaches were presented. Firstly, we utilized widely used classification and regression algorithms. SVM reached the best results. We also proved that data about social behavior of students improve the predictions for a quarter of courses. This approach can be beneficially utilized for the grade prediction of courses with a small number of students.

The second novel approach utilized collaborative filtering techniques and predicted grades based on the similarity of students' achievements. The advantage of this approach was that each university information system stores the data about students' grades which were needed for the prediction unlike the data about students' social behavior. We also succeeded in identifying course dependencies. Finally, we were able to predict the final grades of the investigated course by examining grades of only 3 other courses. The approach can be beneficially used for the grade prediction of mathematical courses.

We also identified groups of courses that are hardly predictable: courses with a different specialization than usual at the Faculty of Informatics, and also large informatics courses which are easy to pass. Finally, we were able to predict the final grade with the error of only one degree in the grade scale for the rest of courses. Half of students' failures were also correctly identified even if the task was difficult due to the fact that all unsuccessful grades constitute less than a quarter of all grades.

7. REFERENCES

[1] Bydžovská, H and Popelínský, L 2 4 The Influence of Social Data on Student Success Prediction. *Proceedings of the 18th International Database Engineering & Applications Symposium*, pp.374-375.

[2] Bydžovská, H 2 5 Are Collaborative Filtering Methods Suitable for Student Performance Prediction? *Progress in Artificial Intelligence - 17th Portuguese Conference on Artificial Intelligence*, pp. 425-430.

[3] Carrington, P., Scott, J. and Wasserman, S. 2005. Models and methods in social network analysis. Structural analysis in the social sciences. Cambridge University Press.

[4] Koprinska, I., Stretton, J., and Yacef, K. 2015. Students at Risk: Detection and Remediation. *The 8th International Conference on Educational Data Mining (EDM 2015)*, pp. 512 – 515.

[5] Manouselis, N. and Drachsler, H. and Vuorikari, R. and Hummel, H. and Koper, R. 2011. Recommender Systems in Technology Enhanced Learning, Recommender systems Handbook Spriger Verlag 2011, pp 387-415.

[6] Marquez-Vera, C. Romero, C. and S. Ventura. 2011. Predicting school failure using data mining. *In Proceedings of the 4th International Conference on Educational Data Mining (EDM'11)*, pp. 271-276.

[7] Matuszyk, P., and Spiliopoulou, M. 2014. Hoeffding-CF: Neighbourhood-Based Recommendations on Reliably Similar Users. *In User Modeling, Adaptation, and Personalization*, volume 8538 of Lecture Notes in Computer Science, Springer International Publishing.

[8] Murtagh, F. and Contreras, P. 2012. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97.

[9] Nghe, T. N., Janecek, P. and Haddawy, P. 2007. A comparative analysis of techniques for predicting academic performance. *37th ASEE/IEEE Frontiers in Education Conference*, Milwaukee, WI 2007.

[10] Nižnan, J , Pelánek, R , and Řihák, J 2 5 Student Models for Prior Knowledge Estimation. *The 8th International Conference on Educational Data Mining (EDM 2015)*, pp. 109-115.

[11] Nooy, W., Mrvar, A. and Batagelj V. 2011. Exploratory Social Network Analysis with Pajek. Structural Analysis in the Social Sciences. Cambridge University Press.

[12] Strecht, P., Cruz, L., Soares, C., Mendes-Moreira, J., and Abreu, R. 2015. A Comparative Study of Classification and Regression Algorithms for Modelling Students' Academic Performance. *The 8th International Conference on Educational Data Mining (EDM 2015)*, pp. 392-395.

[13] Vandamme, J.P., N. Meskens and J.F. Superby, 2007. Predicting academic performance by data mining methods. *Educ. Econ.,* 15, 405-419.

[14] Ventura, S., Romero, C., López, M.-I., and Luna, J.-M. 2013. Predicting students' final performance from participation in on-line discussion forums. *Computers & Education* 68, 458-472.

Course Enrollment Recommender System

Hana Bydžovská
CSU and KD Lab Faculty of Informatics
Masaryk University, Brno
bydzovska@fi.muni.cz

ABSTRACT

One of the main problems faced by university students is to create and manage the semester course plan. In this paper, we present a course enrollment recommender system based on data mining techniques. The system mainly helps with students' enrollment decisions. More specifically, it provides recommendation of selective and optional courses with respect to students' skills, knowledge, interests and free time slots in their timetables. The system also warns students against difficult courses and reminds them mandatory study duties. We evaluate the usability of designed methods by analyzing real-world data obtained from the Information System of Masaryk University.

Keywords

Course enrollment recommender system, student performance, prerequisites, university information system.

1. INTRODUCTION

Recommender systems can be used in different fields including educational environment. Such systems are mainly focused on providing high educational standard and try to enhance the process of teaching and learning [13]. They help with searching for suitable web resources [8], recommend good solutions to improve students' knowledge [4], or analyze data obtained from quizzes and provide a feedback to instructor to modify a quiz [9].

Nowadays, researchers also try to improve personalized searching for beneficial courses. The aim of several projects was to select courses in order to obtain good exam results [12] or recommend elective course modules based on previous students' enrollments using collaborative filtering techniques [6]. Other option is to utilize association rules [1] or ant colony optimization [11].

In the last few years, recommendations became more complex. Besides selecting passable courses, it is essential to recommend beneficial courses [3]. The suitability of courses was determined by the importance in all fields of the university, the ratio of connectivity among courses and by the importance in the student's field of study. Association rules were utilized for searching relationships between courses. Another approach was presented in [7]. To graduate, all defined blocks of courses must be completed by finishing a pre-defined number of courses. They utilized a flow algorithm to find the minimal set of courses that students have to pass.

In this paper, we present a pilot version of the course enrollment recommender system designed at the Faculty of Informatics Masaryk University. All methods were validated on data originated from the Information System of Masaryk University (IS MU). The data contain information on courses, templates defining the mandatory and selective courses, students, study-related attributes, and social behavior data. The designed methods predict students' final grades and recommend them interesting courses with respect to their skills, interests, and free time-slots in the timetable.

2. COURSE ENROLLMENT RECOMMENDER SYSTEM

2.1 Motivation

All students have to follow the obligations and principles stated by their university. Especially at the beginning of the study, it is hard for students to cover all the mandatory duties. At Masaryk University, all semesters are preceded by a course enrollment process. All active students have to enroll a sufficient number of courses to achieve at least the minimal pre-defined amount of credits. If they do not reach the minimum limit, they cannot proceed to the next semester. Students have to pass many courses before finishing their studies successfully. All mandatory courses must be completed. Students have to also pass several selective and optional courses. Analyzing the enrollment statistics, we found out that students prefer interesting and passable courses. Universities usually offer a large number of courses and it is difficult for students to be familiarized with all of them. They are forced to search through the entire course catalog, read many abstracts and syllabi, and compare a large amount of success rate statistics. Naturally, they often discuss courses with other students who have their own personal experiences. Obviously, the decisions they have made during the course enrollment process could significantly influence the whole study progress and the final result.

2.2 System Overview

The current version of the recommender system monitors the number of credits of enrolled courses to ensure successful progression to the next semester. It also reminds them to enroll all mandatory courses. Selective and optional courses are recommended according to the student's performance and interests with respect to free time slots in students' timetables. The system clarifies the decisions to students using notifications. The system also warns against enrolled courses that usually cause problems to students with similar characteristics. If the system identifies a difficult course in the student's enrollment, it informs the student about the potential issue. It allows students to focus more on this course or to revise the enrollment decision. Students can also assess each recommendation whether the recommended courses were interesting and adequately difficult. Based on these assessments, the recommendation algorithms will be modified in order to enhance the relevance of the further recommendations.

3. COURSE TEMPLATES

At our university, templates represent tree-like definitions of mandatory and selective courses for each field of study. The system allows checking the requirements that a student has already accomplished. The completed courses/nodes are marked with a green ring (o) and the uncompleted courses/nodes are marked with a red cross (x).

We examined 67 templates defining the study requirements for active students in the years of 2010-2013 at Faculty of Informatics. An example of a template can be seen in Figure 1.

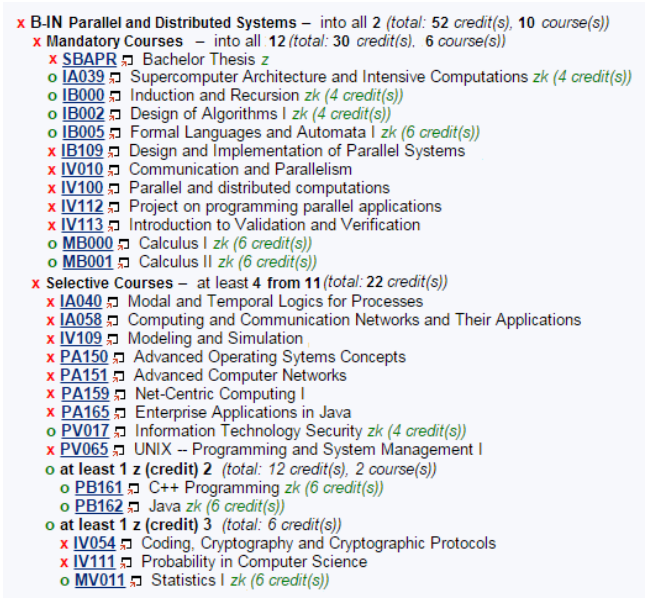


Figure 1. Template of mandatory and selective courses

However, the structure of the templates is often more complicated. Each node defines how many child nodes have to be completed (all, defined by the number of credits, or defined by the number of children). The template does not enforce in which semester courses should be enrolled.

3.1 Which courses do students have to pass before enrolling a certain course?

Some courses have prerequisites that define what a student must meet before he or she can enroll in a certain course. At our university, prerequisites are composed of terms $p_1 \ldots p_n$ that are associated with logical operators AND(&&), OR(||). A term p_i can be a course or a compound term. Prerequisites can be transformed into the template subtree by the following rules:

- p_i && p_j → new node containing p_i and p_j with the rule of fulfillment: all nodes
- p_i || p_j → new node containing p_i and p_j with the rule of fulfillment: at least one of nodes

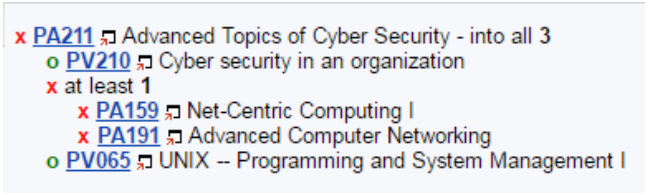


Figure 2. PA211 prerequisites: PV210 && (PA159 || PA191) && PV065

Example of such transformation can be seen in Figure 2. Each template could be extended by prerequisites courses for students to be able to count on them when creating their study plans.

3.2 When do students have to enroll a certain course?

Students can decide in which semester they enroll in a certain course. All graduate students that completed the template requirements were selected and the semester in which the most of them enrolled in the particular mandatory course was calculated by Algorithm 1. Therefore, we remind courses in the proper semesters with respect to students' completed semesters.

Algorithm 1. Semester Selection

Function select_semester(course, template):

sem_max = {sem ∈ semesters | ¬∃sem_2: number_students (sem_2, course, template) > number_students (sem, course, template)}
if (|sem_max| == 1) then
    return sem_max[0];
else if (|sem_max| > 1) then
    return min(sem_max);
else
    return 1;
end if;

Function number_students(semester, course, template):

return the number of students having completed the given template enrolled in the given course in the specific semester;

3.3 Which courses are passable for a certain student?

We focused on the problem of predicting the final grade at the beginning of the semester with the emphasis on identifying unsuccessful students. The following grade scale was used: 1 (excellent), 1.5 (very good), 2 (good), 2.5 (satisfactory), 3 (sufficient), 4 (failed or waived). The value 4 represents students' failure; the others represent a full completion.

We present two different approaches in [2]. Both approaches are validated on 138 courses which were offered to students of the Faculty of Informatics of Masaryk University between the years of 2010 and 2013. The first approach is based on classification and regression algorithms that search for patterns in study-related data and also data about students' social behavior. We prove that students' social behavior characteristics improve prediction for a quarter of courses. The second approach is based on collaborative filtering techniques. We predict the final grades based on previous achievements of similar students. We also present the novel approach how to find out which approach is better for which courses. Finally, we are able to correctly identify half of all failures (that constitute less than a quarter of all grades) and predict the final grades only with the error slightly higher than one degree in the grade scale.

Due to the prediction error, we decided to lower the granularity of predictions to the following three classes: excellent (1, 1.5), good (2, 2.5), and bad (3, 4) to prevent the recommendation of difficult courses. As it can be seen in Table 1, the mean absolute error was below 0.5 and due to the high value of sensitivity the most of unsuccessful students were revealed.

The approaches are beneficially utilized in the presented course enrollment recommender system to warn students against difficult courses and to recommend only passable optional courses. Courses with predicted grade better than bad grade are considered as passable for a student.

Table 1. Prediction Evaluation on Test set

Task	MAE	Sensitivity
Grade prediction	0.609	0.436
Excellent / good / bad prediction	0.474	0.899

4. SELECTIVE COURSES

Students can select different sets of selective courses from the template with respect to their skills and the course content. They have to select enough courses to fulfill the node requirements. We were interested in the student behavior, e.g. information about the most preferred courses.

4.1 Designed Recommendation Methods

We defined a course *c* for a student *a* as interesting by the following function:

$$f(a, c) \begin{cases} 1 \text{ if the student } a \text{ attended course } c \text{ or marked it as favorite} \\ 0 \text{ otherwise} \end{cases}$$

This characteristic defined the student's interest in the course. Therefore, each student can be characterized by a set of his or her interesting courses. We designed the following 4 algorithms to recommend courses:

S1. The most selected courses by students with the same template. We were interested in the student behavior, e.g. information about the most preferred courses. We computed the most frequent path of graduate students in the template. We were inspired by a simple ant colony algorithm and marked each node with the number of students that passed through. The path was computed by universal path finding Algorithm 2.

S2. Courses enrolled by similar students. We calculated the similarity between sets of interesting courses for each student and all graduate students that already completed the template. We utilized Jaccard's coefficient. For each student, we selected the most similar students and recommended their courses. We were searching for the proper size of the neighborhood and evaluated $n \in [1; 25]$. When we sorted the courses in the list by their frequency of occurrence in similar student's lists, we also explored how many of them were suitable to be recommended. We examined $x \in [1; 10]$.

S3. Courses taught by favorite teacher. Students' interesting courses were examined and favorite teachers were revealed. We considered all course lecturers and only student's tutors. The teacher's popularity was defined as the sum of all his or her courses which were considered as interesting. Considering the teacher's popularity, we recommended other teacher's courses if his or her popularity was above the threshold (2).

S4. Courses enrolled by friends. We examined students' social behavior characteristics and their mutual cooperation. We focused on statistical data that represented the interaction among students: explicitly expressed friendship, posts and comments in discussion forums, e-mails statistics, publication co-authoring, or files sharing. This information served as the basis for computing social ties among students by means of a sociogram [2]. From this sociogram, we were able to reveal friends ties among students. We recommended courses that friends considered as interesting and belonged to the template.

The algorithms also observed the following rules:

- Courses recommended for a particular student were limited to courses that should be enrolled in the certain semester:
  $$course's\ semester \quad student's\ semester$$
  Student's semester was defined as the number of commenced semesters and the course's semester was defined as the semester in which other students usually enrolled in the course calculated by Algorithm 1.
- We also did not recommend courses that belonged to the subtree of the template which students had already completed.
- Only courses that could be enrolled in the actual semester were recommended.

Algorithm 2. Finding Path in Template

```
Function process_node (node, template, student):
children ← children of the node;
for each child in children do
        unless (child_computed) then
                process_node(child, template, student);
        end if;
end for;
path; # calculated path
sort children in descending order by the value in the node;
for each child in children do
        path ← child;
        if (node_fulfilled(node, student)) then
                return path;
        end if;
end for;

Function node_fulfilled (node, student):
if (the given node is fulfilled by the given student) then
        return true;
else
        return false;
```

4.2 Recommendation Methods Evaluation

We can assume that students are familiar with the offer of selective courses. Therefore, offline experiments [10] can be suitable approach to evaluate previously mentioned algorithms. All students that enrolled in the semester autumn 2014 and did not complete their templates were selected: 1,444 students in total.

4.2.1 Settings for the algorithm S2

Firstly, we had to evaluate suitable settings for the algorithm S2. Our task was to select suitable courses for students and subsequently detect if they enrolled in them or not. Therefore, the suitable evaluation metrics were precision and recall. To find a balance between precision and recall, the F1 score was also calculated.

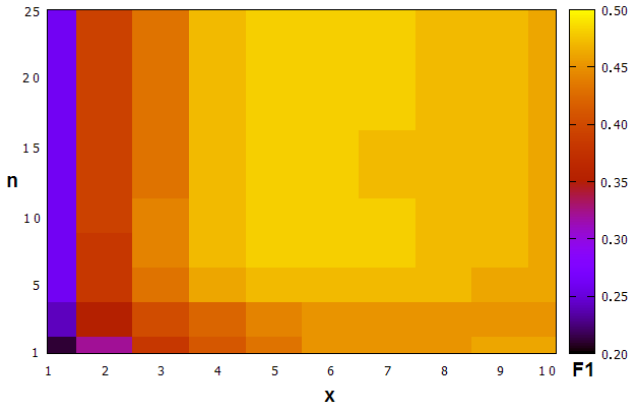


Figure 3. Relationship among the size of the neighborhood *n*, number of selected courses *x* and the value of F1 score

We selected 90% of examined students and calculated the F1 score of the recommendations. Figure 3 shows the relationship among variables *n, x*, and the value of F1. Based on these findings, the following setting was selected for algorithm S2 as the most suitable: $n = 8, x = 5$. This conclusion was also verified on the test set (the rest 10% of students).

4.2.2 All algorithms' evaluation

We utilized all previously described algorithms to recommend courses for each student. The coverage determines the percentage of students for whom we were able to recommend at least one course.

Table 2. Results of selective courses recommendation

Algorithm	S1	S2	S3	S4
Coverage	0.97	0.63	0.60	0.54
Offered courses	2.97	4.81	3.85	4.43
Enrolled courses in the semester autumn 2014	1.63	2.08	1.81	1.88
Enrolled courses anytime	2.82	3.15	2.49	2.85
Precision	0.81	0.56	0.48	0.47
Recall	0.55	0.42	0.28	0.39
F1	0.66	0.48	0.35	0.43
Rank	**1**	**2**	**4**	**3**

The coverage of approaches differs as it can be seen in Table 2. S1 covered almost all students. In contrary, the rest of approaches recommended courses for only 60% of selected students. The average number of courses offered by each algorithm can be seen in the second row. Algorithms recommended 3-5 courses on average. The average number of courses that students really enrolled in autumn 2014 can be seen in the third row. Because the university does not define when students have to enroll courses, we extend the searching for enrollment also to the next semesters. The average number of courses that students really enrolled anytime from autumn 2014 till now can be seen in the fourth row. As it can be seen, the number of enrolled courses almost doubled in all cases. Finally, we also calculated precision and recall for all algorithms. The algorithm S1 reached the best results.

4.2.3 Which courses are selected the most often?

H1: We supposed that students select easier selective courses.

For finding the easiest way to complete the template, we assessed each course using its success rate (the percentage of successful students to all students in the course). However, we had to penalize courses with a small number of students and also the courses with smart students only (with excellent average grade). Therefore, the adjusted success rate (ASR) was defined as:

$$ASR = CSR \quad \log_4 ESAG \quad \frac{NES}{\text{MAX\_ENR}}$$

where CSR defined the course success rate, ESAG defined the average grade of enrolled students, NES defined the number of enrolled students in a course, and MAX_ENR was a constant for the template and defined the maximum number of students enrolled in any course from the template. We calculated the minimal adjusted success rate of courses that have to be passed in the subtree for each node of the template. Subsequently, we employed the Algorithm 2 that selected the easiest courses till the node requirements were met.

For each template $t \in T$ we constructed the easiest path (EP) and also the most frequented path (MFP). Both paths can be represented as a set of selected courses on the path. Jaccards' coefficient (JC) was calculated to compare these sets of courses. The similarity of paths was 0.8 on average for all templates.

$$\frac{\sum_{t \in T} JC(EP, MFP)}{|T|} = 0.8$$

H1 was confirmed. Correlation of EP and MFP over all templates confirmed our hypothesis that students usually select easier selective courses.

5. OPTIONAL COURSES

To fulfill all study requirements, students have to obtain the pre-defined number of credits in their studies. Except credits obtained from mandatory and selective courses, they have to select optional courses. Optional courses for each student were defined as courses that do not belong to the student's template.

5.1 Designed Recommendation Methods

We utilized the same methodology as described in Section 4 for recommendation of selective courses. The main difference was that algorithms did not restrict courses from templates. The courses recommended by algorithms were limited to only passable courses (the predicted grade was not bad) according to the method introduced in Section 3.3.

S1. The most selected courses by students with the same field of study. All optional courses of all students of a certain field of study were selected. The number of students that were interested in each course was calculated and the sorted list of all courses based on the calculated value was created from the most interesting.

S2. Courses enrolled by similar students. We computed the student similarity with all active students and also students graduated in the last five years. The revealed courses were sorted into a list by the number of occurrences in similar students' sets of optional courses.

S3. Courses taught by favorite teacher. Courses were sorted into a list in decreasing order by the popularity of a teacher.

S4. Courses enrolled by friends. Courses were sorted into a list by the number of occurrences in friends' sets of optional courses.

5.2 Recommendation Methods Evaluation

As a contrary to the selective course recommendation, we supposed that students are not familiarized with all the optional courses. Therefore, the offline experiments were not sufficient evaluation technique in this case and we had to conduct a user study [10]. We contacted only selected group of students to request them to assess our recommendations.

We could approach 607 students enrolled in one of our courses in the last semester. Considering the number of students and expecting the lower response rate of students, we selected 5 top rated courses by each algorithm for each student. The coverage of approaches when the algorithm found at least one course to offer is presented in Table 3 in the first row. Only for a half of students, we revealed friends who could inspire students with interesting courses. The average number of offered courses by each algorithm can be seen in the second row. The approach which uses social ties (S4) offered only 4 courses on average.

In our experiment, we offered 10 courses at maximum selected using the 2 our algorithms S_i and S_j for each student. We sorted the students in the list by their average grade in order to be

independent of students' characteristics and nearly randomly selected 2 algorithms that offered its top 5 courses each at maximum to students. We balanced the number of occurrence of each algorithm due to the low coverage of S4. We also merged the list of courses of S_i and S_j in order to not prioritize one of them in the following order: S_{i1}, S_{j1}, S_{i2}, S_{j2}, S_{i3}, S_{j3}, S_{i4}, S_{j4}, S_{i5}, and Sj_5. When both algorithms selected the same course, the course appeared only once in the list. The assessment of the course was added to results for both algorithms.

Table 3. Algorithms coverage

Algorithm	S1	S2	S3	S4
Coverage	1	1	0.96	0.49
Offered Courses	4.98	4.98	4.47	4.02

Subsequently, students were asked for assessing the recommendation during their course enrollment process to increase the possibility of their reaction. Students could mark courses using the following attributes: like, do not like or leave it unanswered. Overall, 172 students responded. The most of them responded in one week since the invitation (see Figure 4).

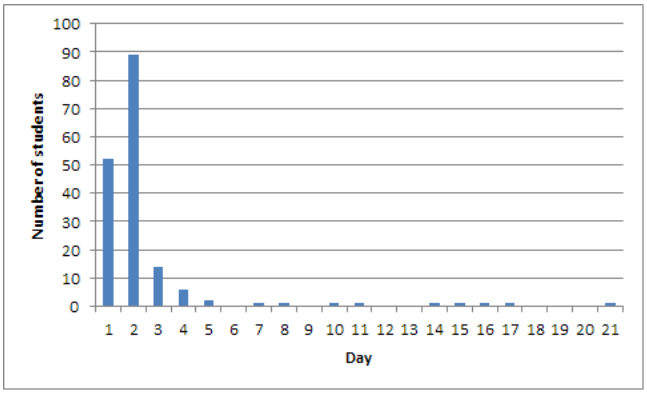


Figure 4. Students' reaction period

The distribution of students' reactions is shown in Figure 5. The best recommendation was offered by the algorithm S2. The algorithm is based on the similarity of students' sets of interesting courses.

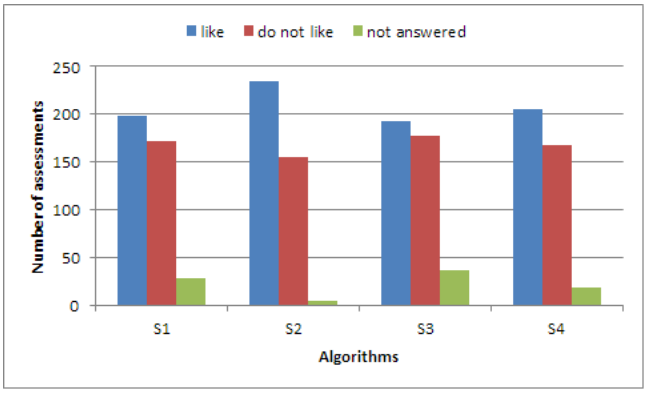


Figure 5. Assessed courses

The number of students assessed (NSA) our algorithms was almost in balance. Each student was included twice: for each of algorithms that assessed. As it can be seen in Table 4, we obtained more assessments of courses inspired by friends' selections (S4). It can mean that students with more social ties in the system are more active. We omitted recommendations that were not assessed. For all algorithms we obtained enough assessments to be able to properly evaluate them. We utilized the same evaluation metrics as for selective courses besides recall because we could not compute false negatives. On average for all algorithms, students liked 2-3 of 4-5 offered courses.

Table 4. Algorithms evaluation

Algorithm	S1	S2	S3	S4
NSA	79	79	82	99
Liked Courses	2.52	2.97	2.35	2.07
Offered Courses	5	5	4.8	3.9
Precision	0.53	0.60	0.52	0.55
Rank	**3**	**1**	**4**	**2**

Considering all evaluation methods, we determined the ranking of algorithms' success rate. Algorithm based on similarity of interesting courses (S2) reached the best results. However, the final solution will combine all algorithms to achieve best results.

6. RECOMMENDATIONS

We have designed new elements for Registration Application which might be available to all students of Masaryk University in the future. The first enhancement presents the predicted difficulty of courses to students. The predictions are computed by the method described in Section 3.3. The predicted grades correspond to the following color:

- x cellent grade green color.
- ood grade yellow color.
- Bad grade red color.

All predictions are presented as the icons of corresponding color. When we have no predictions, there is no icon. We try to predict grades of courses that students enrolled or courses that we recommend to them (see Figure 6). Based on these warnings, students can concentrate on difficult courses or revise their choices depending on the planned workload in the semester.

The second improvement is the panel on the right (see Figure 6) where the recommended courses are presented. For each student we remind mandatory courses, recommend selective and optional courses selected by methods introduced in Sections 4 and 5, and also recommend their prerequisite courses. After clicking the wrench icon, the short explanation of each recommendation is displayed to increase students' trust to the system [5]. They can also assess each recommendation. Based on assessments we continuously improve our algorithms.

7. CONCLUSION

We presented a pilot version of course enrollment recommender system that reminds students their duties, warns them against difficult courses and recommends them potentially beneficial courses. Therefore, the system helps students with their decisions during the enrollment process at the beginning of each semester.

More specifically, we designed four algorithms suitable for the course recommendation. The first algorithm searches for the most frequently enrolled courses. The second algorithm utilizes similarities of students based on courses of their interests. The third algorithm recommends courses of students' favorite teachers. The last algorithm calculates the social ties among

students and selected courses which were interested for students' friends.

The most suitable algorithm for the selective course recommendation was the first described algorithm. Students usually selected easier courses defined in their templates. In contrary, the best results for the optional courses recommendation achieved the second algorithm utilizing students' similarities. However, we decided to employ all methods in the system due to the high students' satisfaction with recommendations. Optional courses were also recommended only if we predicted that students could pass the course and they had free time slots in the timetable for the course. We validated all designed methods on data originated from students of the Faculty of Informatics Masaryk University stored in the university information system.

We also introduced the environment that presents recommendations to students, offers them the explanations why the courses were selected, allows them to leave a feedback, warns them against difficult courses, and reminds them important events that should be accomplished, e.g. enroll in mandatory courses or enroll enough credits. The designed course enrollment recommender system will be a part of the university information system in the future.

8. REFERENCES

[1] Bendakir, N. and Aimeur, E. 2006. Using Association Rules for Course Recommendation. In *Proceedings of the AAAI Workshop on Educational Data Mining,* pp. 31-40.

[2] Bydžovská, H. 2016. A Comparative Analysis of Techniques for Predicting Student Performance. *Proceedings of the 9th International Conference on Educational Data Mining 2016* (Accepted).

[3] Lee, J. Ch. Y. and Lee, K.-W. 2011. An intelligent course recommendation system. *Smart Computing Review*, 1(1).

[4] Loll, F. and Pinkwart, N. 2009. Using collaborative filtering algorithms as elearning tools. *In Proceedings of the 42nd Hawaii International Conference on System Sciences*.

[5] O'Donovan, J. and Smyth, B. 2005. Trust in Recommender Systems. *Proceedings of the 10th International Conference on Intelligent User Interfaces*, pp. 167-174.

[6] O'Mahony, M. P., and Smyth, B. 2007. A recommender system for on-line course enrolment: an initial study. *In Proceedings of the ACM conference on Recommender systems (RecSys '07)*, pp. 133–136.

[7] Parameswaran, A., Venetis, P., and Garcia-Molina, H. 2011. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Trans. Inf. Syst.*, 29(4):20:1–20:33.

[8] Recker, M. M., Walker, A., and Wiley, D. 2004. Collaborative information filtering: A review and an educational application. *International Journal of Artificial Intelligence in Education*, Volume 14 Issue 1, pp. 3-28.

[9] Romero, C., Zafra, A., Luna, J. M., Ventura, S. 2013. Association rule mining using genetic programming to provide feedback to instructors from multiple-choice quiz data. *Expert Systems* 30(2): 162-172.

[10] Shani, G. & Gunawardana, A. 2011. Evaluating recommendation systems. Recommender Systems Handbook, pp. 257-297.

[11] Sobecki, J. and Tomczak, J. M. 2010. Student courses recommendation using ant colony optimization. In *Proceedings of the Second international conference on Intelligent information and database systems*: pp. 124-133.

[12] Vialardi, C., Chue, J., Peche, J. P., Alvarado, G., Vinatea, B. Estrella, J., and Ortigosa, A. 2011. A data mining approach to guide students through the enrollment process based on academic performance. *User Modeling and User-Adapted Interaction*, Volume 21, Issue 1, pp. 217-248.

[13] Vuorikari, R., Hummel, H., Manouselis, N., Drachsler, H., and Koper, R. 2011. Recommender Systems in technology enhanced learning. In Recommender systems Handbook, pp. 387-415. Spriger Verlag.

Figure 6. Demonstration of Interface

Data-driven Automated Induction of Prerequisite Structure Graphs

Devendra Singh Chaplot
School of Computer Science
Carnegie Mellon University
dchaplot@cs.cmu.edu

Yiming Yang
School of Computer Science
Carnegie Mellon University
yiming@cs.cmu.edu

Jaime Carbonell
School of Computer Science
Carnegie Mellon University
jgc@cs.cmu.edu

Kenneth R. Koedinger
School of Computer Science
Carnegie Mellon University
koedinger@cmu.edu

ABSTRACT

With the growing popularity of MOOCs and sharp trend of digitalizing education, there is a huge amount of free digital educational material on the web along with the activity logs of large number of participating students. However, this data is largely unstructured and there is hardly any information about the relationship between material from different sources. We propose a generic algorithm to use educational material and student activity data from heterogeneous sources to create a Prerequisite Structure Graph (PSG). A PSG is a directed acyclic graph, where the nodes are educational units and the edges specify the pairwise ordering of the units in effective teaching by instructors or for effective learning by students. We propose an unsupervised approach utilizing both text content and student data, which outperforms to supervised methods (utilizing only text content) on the task of estimating a PSG.

1. INTRODUCTION

Students need prior knowledge for thorough understanding of educational content. This need imparts an implicit order in learning educational concepts. Determining this order requires significant human time and effort. Furthermore, relying on expert knowledge to determine this order is subject to inconsistencies due to 'expert blind spot' [8]. We aim to leverage free educational material on the web, and huge amount of student activity logs associated with them, to create a universal Prerequisite Structure Graph (PSG). We define PSG as a directed acyclic graph, where the nodes are the universal concepts in an educational domain and the edges specify the pairwise ordering of concepts in effective teaching by instructors or for effective learning by students. The proposed unsupervised methods utilize both textual content and student performance data to perform better than supervised methods utilizing textual content. They can be generalized to find the learning order between any pair of educational elements from heterogeneous resources, at any level of granularity (courses, units, modules, skills, etc.).

The rest of the paper is divided as follows. The related work pertaining to the proposed methods is discussed in Section 2. Section 3 describes the dataset used for experiments. Performance-based and text-based unsupervised induction of a PSG are described in Sections 4 and 5, respectively. We describe the method of combining text-based and performance-based approaches in Section 6. Experiments and results are presented in Section 7. In Section 8, we analyze whether the concepts extracted by proposed methods are meaningful. Conclusions and future directions are covered in Section 9.

2. RELATED WORK

Currently, the construction of Concept Graphs majorly depends on manual work of domain experts. Recent work by [10] on Concept-Graph-Learning (CGL), focuses on determining the relationship between different University courses and MOOCs by inferring concepts from course descriptions. The proposed methods are completely unsupervised as compared to supervised CGL which requires partial instructor-specified links. One other recent work includes extracting a concept-hierarchy from textbooks [9], where the focus is only on extracting the hierarchies between concepts and the learning is only done at the concept level. We differentiate ourselves from this work with the fact that we learn the prerequisite relationships between educational concepts rather than hierarchies, and our method is generalizable to any granularity of educational elements.

Another indicator of prerequisite links between educational elements is student performance. An early approach to inferring prerequisite graphs from student performance data is knowledge spaces [2], which uses associations between student success on different classes of tasks to infer prerequisite relationships. The essential idea is that if students are highly likely to get tasks of type A correct (e.g., finding least common multiples) conditioned on getting tasks of type B correct (e.g., adding fractions with unlike denominators) but not the other way around (i.e., many students that can find common multiples fail at adding fractions), then A is a pre-

requisite of B. Subsequently, algorithms for inferring cognitive models of student learning from data have been developed and it is possible to infer prerequisite relationships from the results of these models [1]. The methods we propose are different as we utilize not only the student performance data, but also student activity data along with large amounts of text in course material. Also, previous approaches assume that there is no learning between attempts at different problems, which is suitable for standardized testing scenario but not true for student performance logs of complete courses.

3. DATASET

We use the text content and student activity and quiz performance data from Georgia Tech's "Introduction to Psychology" MOOC which uses content from the Open Learning Initiative of Carnegie Mellon University [6]. The course spans over 12 weeks and a major topic of Psychology (like intelligence, personality, psychological disorders, etc.) was covered in each week of class. For each week, the text content from the corresponding unit(s) was extracted. The unit(s) covered in each week are shown in Table 1. On an average, each unit contained 12545 word instances with a standard deviation of 3730. For simplicity, we will use Unit i to denote the content covered in Week i, although the content covered in week i might include multiple units in the course. Besides the text inside course units, we also used text in the weekly quizzes separately to evaluate our text-based methods.

The course also contained ungraded practice activities within each unit. At the end of each week (from week 1 to week 11), students were assessed by a high stakes quiz containing questions from content covered in the corresponding week. The dataset includes the number of interactive activities and quiz scores of 1154 students for each week.

This dataset is ideal for our analysis since it has both the textual data of course material and the student activity and quiz performance data. We aim to predict prerequisite links between weeks using this data, which will imply prerequisite links between corresponding units. For example, a prerequisite link from Unit 9 to Unit 11 implies prerequisite link from Personality to Disorders, or in other words, a student who has learned Personality will be better able to learn Disorders.

For evaluation, the dataset was first annotated by three non-experts who determined whether a prerequisite link exists between content covered in any two units. If a prerequisite link exists from Unit i to Unit j, we call it a positive link, and conversely, if there is no link, it is called a negative link. The average percentage agreement for positive links between each pair of annotator was 29.6% while the percentage agreement for positive links among all the annotators was 18.7%. Since the inter-annotator agreement was very low, we got the dataset annotated by a domain expert. All the links marked positive by all three non-expert annotators were also marked positive by the domain expert, except one link. Finally, we took 15 links marked positive and domain expert, and 1 more link marked positive by all non-expert annotators as the set of positive links. Therefore, among 110 possible links, 16 links were labeled as positive and rest negative. Note that 55 out of 110 possible links are backward (i.e. from Unit i to Unit j such that $i > j$), which should be implicitly negative, but we will not use the information about the ordering of units in any of the proposed methods so that our methods are generalizable to any pair of educational elements: modules, chapters or whole courses.

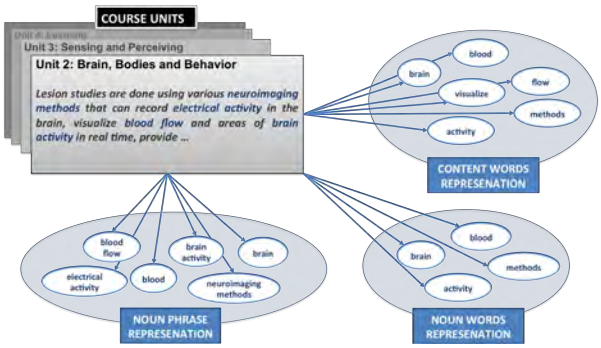


Figure 1: Example of three types of concept space representation schemes: Content Words, Noun Words and Noun Phrases.

Week	Unit(s) Covered
1	Introduction and Methods
2	Brains, Bodies, and Behavior
3	Sensing & Perceiving
4	Learning
5	Memory
6	Language and Intelligence
7	Lifespan development
8	Emotion and Motivation
9	Personality
10	Psychology in Our Social Lives
11	Disorders

Table 1: Unit(s) covered in each week of "Introduction to Psychology" course.

4. TEXT-BASED METHODS

Each educational unit consists of a set of canonical educational concepts. The text content in each educational unit can be used to find the concepts involved in it. The set of concepts in all units is defined as the universal concept space [10]. We define three concept space representation schemes as follows:

- **Content Words Representation (Word)**: The set of content words (Nouns, Verbs, Adjectives and Adverbs) occurring in the course content is used as the concept space. The words are lemmatized using MIT Java Wordnet Interface (JWI) [3].
- **Noun Words Representation (Noun)**: In this representation scheme, we only use set of nouns occurring in the course content as the concept space rather than all content words. These are again lemmatized using MIT JWI.
- **Noun-Phrase Representation (NP)**: In this representation scheme, the set of noun phrases (of depth less than 5) occurring in the course content is used as the concept space.

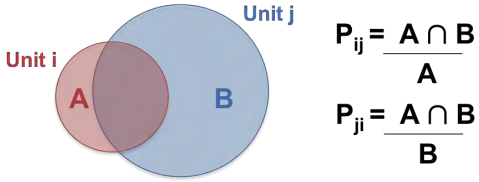


Figure 2: Overlap Method

An example of these three types of representation schemes is shown in Figure 1. The Concept space can be represented using other schemes such as Sparse Coding of Words and Distributed Word Embedding, but these produce latent concepts, which are not human understandable. Furthermore, previous results indicate word-based Representation scheme is more effective than latent concept based representation schemes [10].

Let the total number of the concepts in the concept space be p. Then the educational content in each unit can be represented by a p-dimensional vector, where each element is the frequency of corresponding concept (word, noun or noun-phrase) in the text content of the unit. The concept frequency can be normalized using the following quantities:

- **Collection Frequency (CF):** Total number of occurrences of the word in the collection or in our case, course. This normalizes concept frequencies such that all concepts are given equal weightage.
- **Document Frequency (DF):** Number of documents or in our case, units, that contain the concept. This gives less weightage to words occurring in most units such as module, learning objective, psychology, etc.
- **Wordnet Frequency (WF):** The frequency of word given in WordNet which represents the frequency of word in naturally occurring domain-independent text. This re-scales the frequencies such that domain-specific psychology terms have more weightage than generic terms.

We first describe an unsupervised method which determines prerequisite links based on only the text overlap between educational units. The key idea is that course unit u_i is a prerequisite of u_j to the extent that u_i is a probabilistic subset of u_j (i.e., most concepts involved in u_i are mostly involved in u_j) and u_j is not a probabilistic subset of u_i (i.e., most concepts involved in u_j are not involved in u_i). This idea of using asymmetry in computing the probabilistic subset is motivated by the theory of knowledge spaces [2], but we use text information rather than performance data.

Let x_i be a vector denoting the concept space representation of unit u_i. The length of this vector is the total number of the concepts. Each element of this vector is the frequency of the concept in the unit or one of the normalized versions of concept frequency (CF, DF or WF). The intuitive gloss on how we compute the probability that x_i is a probabilistic subset of x_j is by dividing the size of the intersection of x_i and x_j by the size of x_i (A is a subset of B if $A \cap B = A$ and less so to the extent that $A \cap B < A$, see Figure 2). Mathematically, we define P_{ij} as the ratio of sum of elements of pairwise minimum of x_i and x_j to the sum of elements of x_i:

$$P_{ij} = \frac{sum(min(x_i, x_j))}{sum(x_i)} \quad (1)$$

Then P_{ij} is the weight of the prerequisite link from unit i to unit j, which ranges from 0 to 1.

5. PERFORMANCE-BASED METHODS

Our particular approach for unsupervised induction of PSG based on student performance data grows out of recent analysis of student performance [6] which concludes that interactive activities are more indicative of learning gains than video watching or online text reading. In subsequent analysis, it was found that student learning within a course unit is more highly predicted by their activity within that unit than within other units [7]. However, there is an additional learning outcome boost associated with greater activities before a target unit, but not with greater activities after that unit. This result is consistent with there being prerequisite relationships between prior and later units and was the inspiration for new algorithm development on performance-based PSG inference.

The key idea behind the proposed performance-based methods is that more the activity in unit i predicts success in unit j, the more likely is unit i a prerequisite of unit j. This means that if students who do more activities in week i perform better in the week j quiz, as compared to students who do fewer activities in week i, then there is an evidence for a prerequisite link from content in week i to week j. Let

y_j be Quiz Scores in week j,

x_i be the number of interactive activities done in week i, and

w_{ij} be the parameters denoting the effect of activities in week i on quiz in week j, which we want to estimate. The value of the parameter is the strength of corresponding prerequisite relationship.

We define two methods for predicting prerequisite links using student performance data:

- **Correlation:** The effect of activities in week i on the performance in week j is estimated by the correlation between the number of activities by students in week i and the quiz scores of students in week j quiz. Let $\rho(X, Y)$ be the Pearson correlation coefficient between X and Y. Then,

$$w_{ij} = \quad \rho(x_i, y_j) = \frac{cov(x_i, y_j)}{\sigma_{x_i}\sigma_{y_j}}$$

- **Multiple Linear Regression:** We compute a linear regression for student quiz scores across the 11 units of the course where the dependent variable is student quiz score for the target unit and the independent variables are number of activities students do within each unit. Let $\boldsymbol{w_j} = [w_{1j}, w_{2j}, ..., w_{11j}]$, be a vector denoting the effects of activities in all weeks on quiz score of week j. We define multiple linear regression using lasso regularization as follows:

$$\boldsymbol{w_j^*} = \underset{\boldsymbol{w_j}}{\text{argmin}} \quad \sum_n (y_j - \boldsymbol{x^T w_j})^2 + \lambda \|\boldsymbol{w_j}\|$$

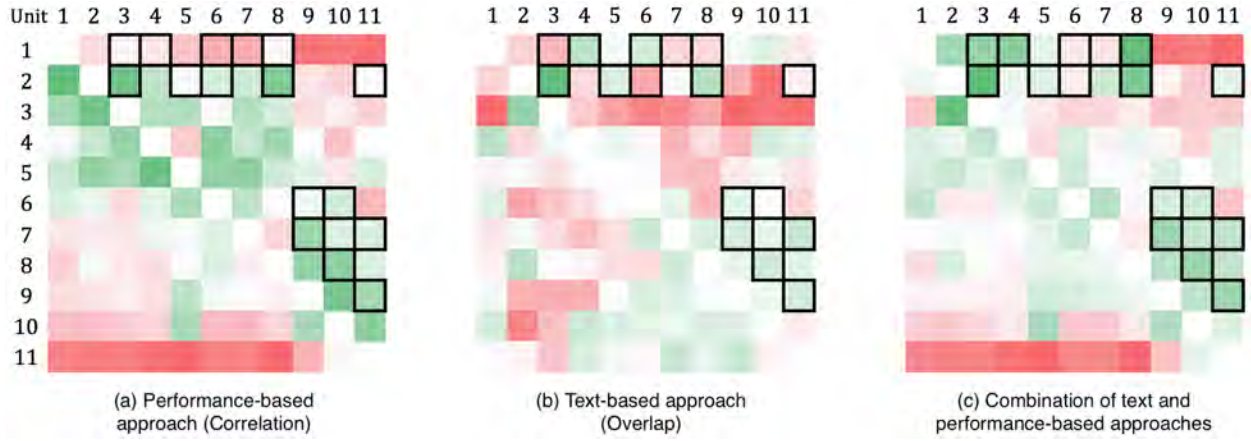


Figure 3: The heat map of strength of links from Unit i to Unit j for (a) Performance-based (Correlation) approach, (b) Text-based (Overlap) approach and (c) Combination of both. The black boxes represent the prerequisite links labeled by domain experts. Note that there is no link from Unit 7 to Unit 10, even though it appears to be surrounded by a black box.

Method Name	Method Type	Data Utilized	MAP	AUC
Regression	Unsupervised	Performance	0.562	0.571
Correlation	Unsupervised	Performance	0.604	0.720
Overlap	Unsupervised	*Quiz* Text	0.693	0.700
Overlap	Unsupervised	*Unit* Text	0.743	0.710
Overlap + Corr	Unsupervised	Performance & *Quiz* Text	0.798	0.820
Overlap + Corr	Unsupervised	Performance & *Unit* Text	**0.837**	**0.840**
CGL[10]	Supervised	*Unit* Text & Labeled links	0.747	0.820

Table 2: Comparison of all methods

6. COMBINING TEXT-BASED AND PERFORMANCE-BASED METHODS

We observed that most of the prediction errors in unsupervised text-based and performance-based methods were due to false-positives. This is because the dataset is imbalanced towards negative class with 85.45% negative labels. Unsupervised systems lacking this information predict positive and negative instance without any prior bias. In order to reduce the errors due to false positives, we propose to predict a positive link only when both methods indicate a positive link.

We get two square matrices of dimension equal to the number of units in the course, one each from text-based and performance-based methods. The $(i, j)^{th}$ element of these matrices represents the weight of the prerequisite link from unit i to unit j obtained from the corresponding method. We combine the two methods by first forcing diagonal entries (self-links) to be 0, then standardizing both the matrices such that both have zero mean and equal variance and then just applying a pairwise minimum over these standardized matrices. This approach predicts a link between any ordered pair of units only if both methods suggest that there should be a link between them. The combination of both methods using a pairwise minimum operation performed better than combination using pairwise summation, pairwise maximum and pairwise product. We also explored more complex models for combination, but found no evidence to justify model complexity.

7. EXPERIMENTS & RESULTS

We gathered and annotated the dataset for experiments as described in Section 3. For evaluation, we used macro-averaged Mean Average Precision (MAP) [5] and Area under ROC Curve (AUC) [4], which are popular metrics in ranked list retrieval and link detection evaluations [10].

The first two rows in Table 2 show the performance of two proposed performance-based methods: Multiple Linear Regression and Correlation. As the Correlation method performed better than Regression method (MAP 0.604 vs 0.562 and AUC 0.720 vs 0.571), we will use Correlation method for combining with text-based methods. The third and fourth column in Table 3 show the performance of text-based Overlap method over different concept space representation types and normalization types. The last two columns of this table show the performance of Overlap method combined with Correlation method. We compare this combined method to supervised Concept Graph Learning algorithm (CGL) [10]. The best results of all methods are summarized in Table 2, which shows that the unsupervised method which combines text-based and performance-based approaches outperforms supervised concept graph learning algorithm by a considerable margin (MAP 0.837 vs 0.747 and AUC 0.840 vs 0.820). As seen Table 3, the combined method performs better than CGL for most concept space representation and normalization types. Note that as compared to supervised CGL method, the proposed method ('Overlap+Corr') utilizes performance data in addition to the text content in educational material but doesn't require labeled links from experts. The results in Table 3 also suggest that on an average, Noun Phrase concept space representation works best for all text-based methods, although there is no clear winner among Normalization types.

Method Name		Overlap		CGL		Overlap+Corr	
Method Type		Text Unsupervised		Text Supervised		Perf+Text Unsupervised	
Rep Type	Norm Type	MAP	AUC	MAP	AUC	MAP	AUC
Word	None	0.656	0.640	0.685	0.789	0.686	0.750
	CF	0.667	0.680	0.742	0.805	0.717	0.800
	DF	0.638	0.660	0.638	0.766	0.836	0.830
	WF	0.693	0.660	0.676	0.781	0.730	0.800
NP	None	0.661	0.680	0.722	0.789	0.745	0.810
	CF	0.703	0.710	**0.747**	**0.820**	0.792	0.820
	DF	**0.743**	**0.710**	0.572	0.773	**0.837**	**0.840**
	WF	0.717	0.710	0.743	0.805	0.748	0.820
Nouns	None	0.734	0.670	0.751	0.805	0.746	0.820
	CF	0.681	0.680	0.687	0.797	0.821	0.810
	DF	0.721	0.680	0.535	0.766	0.755	0.830
	WF	0.738	0.680	0.696	0.797	0.748	0.820

Table 3: Comparison of different concept space representation schemes (Rep Type) and different Normalization schemes (Norm Type) over different text-based methods. CF, DF and WF refer to Collection Frequency, Document Frequency and WordNet Frequency, respectively, as described in Section 4. The best AUC and MAP scores for each method are marked in **bold**.

We analyzed the weights of links predicted by different methods to understand how the combination of text and performance based methods affects our prediction. Figure 3 shows a heat map of strength of links between all pairs of Units. Each $(i, j)^{th}$ element in the matrix represents the strength of link from Unit i to Unit j, where green is denoting higher strength and red is denoting lower. Note that the heat of the colors is determined by relative value of the weights in one matrix and not absolute values across matrices. This is because AUC and MAP metrics evaluate relative value of predicted weights rather than absolute values. The black boxes represent the prerequisite links labeled by experts. The figure indicates that the estimates of performance and text-based approaches compliment each other to give better estimates when combined.

8. DISCUSSIONS

Figure 4 demonstrates a subset of prerequisite links identified by the proposed method and a subset of overlapping concepts occurring in them in the concept space. We would like to analyze whether the concepts identified by the proposed method are meaningful. Consider the relationship between Unit 11, 'Emotion and Motivation' and Unit 13, 'Psychology in Our Social Lives'. All the proposed methods estimate significant weights for link from Unit 11 to Unit 13. Figure 6 shows a part of the concept space representation using Content Words Representation scheme for these units. Overlap method indicates a strong prerequisite link from Unit 11 to Unit 13 due to significant overlap between the concepts in these units. Looking into the contents of these units, the Unit 11, 'Emotion and Motivation' consists of 'Human Motivation' module which involves understanding the motivation behind sexual behavior. It introduces concepts of 'attractiveness', 'proximity' and 'similarity' as motivating factors behind sexual interest. Unit 13, 'Psychology in Our Social Lives' requires the understanding of these concepts in order to understand 'Interpersonal Attraction' in 'Close Relationships' module. Since there are more concepts in Unit 13 like 'personality', 'aggression', 'stimulus', 'judgment', etc. which are not present in Unit 11, $P_{11,13}$ is greater than $P_{13,11}$. Thus, the concepts extracted by the proposed Concept Representation schemes appear to be interpretable and meaningful.

Similarly, we also try to interpret the performance-based results by inspecting the text of the interactive activities within the course. For example, the interactive activities in 'Human Motivation' module correspond to understanding concepts of 'attractiveness', 'proximity' and 'similarity'. The quiz at the end of unit on 'Psychology in Our Social Lives' also contains a question about role of proximity and similarity in interpersonal attraction. Therefore, the students who do more activities in week 8 (involving Unit 11 content) perform better on the week 10 (involving Unit 13) quiz (as compared to students who do fewer week 8 activities) and thus, performance-based approaches identify this relationship. Figure 5 shows the average number of activities of students in prior units as a function of their quiz scores in later unit for set of positive and negative links. The average number of activities in prerequisite units is greater than non-prerequisite units for all quiz scores which is a possible explanation of the effectiveness of performance-based methods. Also, the correlation between number of activities and quiz scores suggests that interactive activities are indicative of learning gains.

9. CONCLUSIONS & FUTURE WORK

We proposed completely unsupervised methods to leverage freely available textual content in educational resources and student performance & activity data for predicting prerequisite structure graph between arbitrary educational resources. Three different concept space representation schemes have been used for text-based methods with a variety of normalization methods for concept frequencies. We also show that when unsupervised text-based and performance-based methods are combined, they supplement each other to outper-

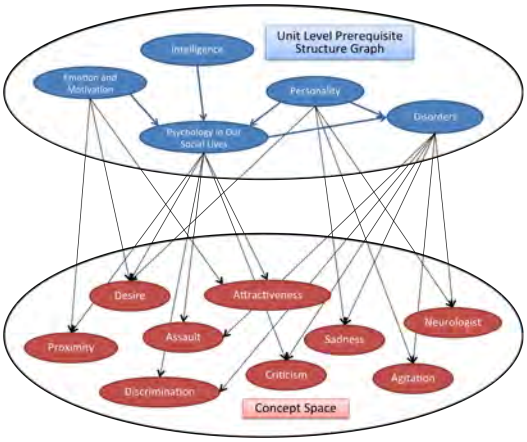


Figure 4: Demonstration of prerequisite links between different units in 'Introduction to Psychology' Course and a subset of overlapping concepts.

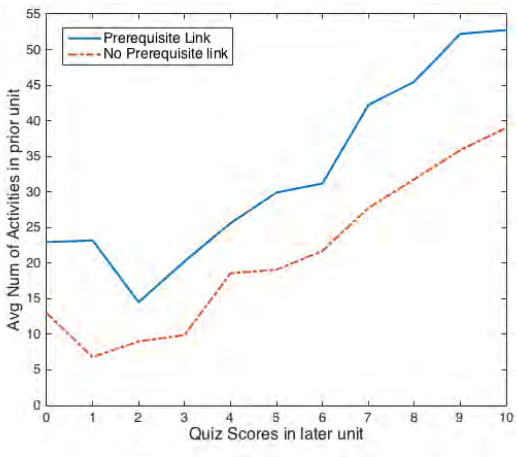


Figure 5: The average number of activities of students in prerequisite units as a function of their quiz scores in post-requisite unit.

form sophisticated supervised methods. Concepts extracted using the proposed representation schemes seem to be interpretable and meaningful from educational perspective.

While the results are encouraging, a limitation of the current work is the size of the dataset. Although the text content in the course and student activity and performance data is rich, the number of positive prerequisite relations in the dataset is low. Validation of proposed methods on diverse educational data from different courses is required to test their generalizability and scalability. Furthermore, conducting a long-term user-study involving students to verify if the predicted prerequisites help them improve their performance over a course, would be useful.

10. REFERENCES

[1] M. C. Desmarais, A. Maluf, and J. Liu. User-expertise modeling with empirically derived probabilistic implication networks. *User modeling and user-adapted interaction*, 5(3-4):283–315, 1995.

[2] J.-P. Doignon and J.-C. Falmagne. Spaces for the assessment of knowledge. *International journal of man-machine studies*, 23(2):175–196, 1985.

[3] M. A. Finlayson. Java libraries for accessing the princeton wordnet: Comparison and evaluation. In *Proceedings of the 7th Global Wordnet Conference, Tartu, Estonia*, 2014.

[4] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[5] K. Kishida. *Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments*. National Institute of Informatics Tokyo, Japan, 2005.

[6] K. R. Koedinger, J. Kim, J. Z. Jia, E. A. McLaughlin, and N. L. Bier. Learning is not a spectator sport: Doing is better than watching for learning from a mooc. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 111–120, 2015.

[7] K. R. Koedinger, E. A. McLaughlin, J. Z. Jia, and N. L. Bier. Is the doer effect a causal relationship? how can we tell and why itâĂŹs important. In *Proceedings of the Sixth International Learning Analytics & Knowledge Conference*, 2016.

[8] M. J. Nathan, K. R. Koedinger, and M. W. Alibali. Expert blind spot: When content knowledge eclipses pedagogical content knowledge. In *Proceedings of the Third International Conference on Cognitive Science*, pages 644–648. Citeseer, 2001.

[9] S. Wang, C. Liang, Z. Wu, K. Williams, B. Pursel, B. Brautigam, S. Saul, H. Williams, K. Bowen, and C. L. Giles. Concept hierarchy extraction from textbooks. In *Proceedings of the 2015 ACM Symposium on Document Engineering*, DocEng '15, pages 147–156, New York, NY, USA, 2015. ACM.

[10] Y. Yang, H. Liu, J. Carbonell, and W. Ma. Concept graph learning from educational data. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 159–168. ACM, 2015.

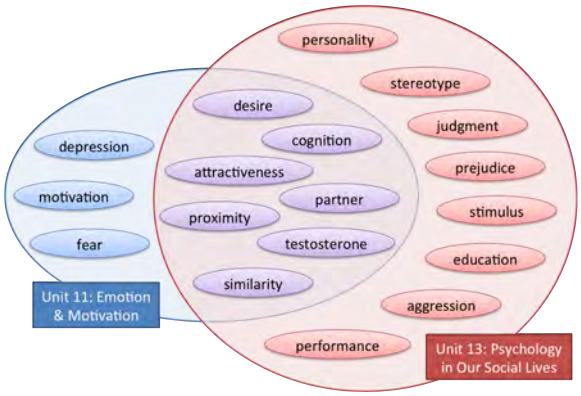


Figure 6: Demonstration of overlap of concepts between units on 'Emotion and Motivation' and 'Psychology in Our Social Lives' and prediction of prerequisite link using Overlap method.

Exploring Learning Management System Interaction Data: Combining Data-driven and Theory-driven Approaches

Hongkyu Choi, Ji Eun Lee, Won-joon Hong, Kyumin Lee, Mimi Recker, Andy Walker
Utah State University
Logan, UT, USA
{hongkyu.choi, jieun.lee, wonjoon.hong}@aggiemail.usu.edu
{kyumin.lee, mimi.recker, andy.walker}@usu.edu

ABSTRACT

This research connects several data-driven educational data mining approaches to a framework for interaction developed in educational research. In particular, 10 million usage data points collected by a Learning Management System used by students and teachers in 450 online undergraduate courses were analyzed with this framework. A range of educational data mining techniques were employed, including K-means clustering, multiple regression, and classification, to both explore and predict student final grades and course completion rates. Findings show that support for the overall model varied with the way data were mapped to the framework (e.g., static vs. temporal features) and the analysis technique used (with clustering and classification providing more useful insights).

Keywords

Learning Management System, Interactions in Online Learning, Clustering, Prediction

1. INTRODUCTION

Educational data mining (EDM) studies have typically relied upon data-driven techniques in order to extract useful patterns and information from large-scale educational datasets [11]. While these data-driven approaches have provided important contributions, some have argued that their inherent a-theoretic nature may fall short in terms of providing insight into the development of educational theory and practice [6]. As such, more studies are needed that better connect EDM findings to educational theory, research, and practice.

To address this need, this paper integrates a theory-driven approach with a data-driven approach to explore student learning outcomes, activities, and patterns as they interact with course content using a popular Learning Management System (LMS), called Canvas. Specifically, for the theory-driven approach, we apply an interaction framework [2] to explore how patterns in the LMS data are related to student final grades and course completion rates at a course level – a macro-perspective. Here, we use K-means clustering and multiple regression analysis. For the data-driven approach, we build classifiers based on machine learning algorithms to predict a student's final grade and whether a student will complete a course or not, providing a micro-perspective.

In particular, we conducted three tasks by addressing following research questions: 1) How many clusters of courses are found based on users' interaction patterns? Are there relationships between individual interaction clusters and course features (size, content, level)? 2) Do the interaction patterns significantly predict student final grades and course completion rates? 3) Can we build effective classifiers to predict an individual student's final grade and whether each student will complete a course? Are the pre-built classifiers still robust and effective for the next semester's data? How many weeks in a semester are needed to discover low performing students or non-course completers (i.e., who may drop out a course)?

2. BACKGROUND

2.1 Interaction in Online Learning

Interaction has long been a significant research topic in the field of educational technology. Nonetheless, it remains a hard concept to define, as it is multifaceted and complex [1, 7]. Some researchers have taken a more restrictive view by excluding non-human factors, and focusing only on human interactions [5]. However, others argued that both human and non-human interactions are integral aspects of the educational experience [1, 2, 4]. Further, supporting various combinations of interaction among teacher, student and the content can help foster a community of inquiry in online learning [4].

In particular, Moore [7] categorized interaction into three types: (i) learner-content interaction, (ii) learner-instructor interaction and (iii) learner-learner interaction. Anderson and Garrison [2] expanded Moore's categorization by differentiating between teacher-content and student-content interaction. In their final model, teacher-content (TC) interaction refers to teachers creating content and learning activities. Student-content (SC) interaction refers to students' interactions with various forms of educational content including reading texts, completing assignments, and working on projects. Student-teacher (ST) interaction includes both asynchronous and synchronous communication between students and teachers. Finally, student-student (SS) interaction

Table 1: Characteristics of 450 courses.

Course characteristics		|Courses|	Percent
STEM Non-STEM	STEM	116	25.8%
	Non-STEM	334	74.2%
Course size	Small (<21)	107	23.8%
	Med (<51)	210	46.7%
	Large (51+)	133	29.5%
Course level	1000 level	156	34.7%
	2000 level	79	17.5%
	3000 level	157	34.9%
	4000 level	58	12.9%

refers to interaction between individual students.

There have been several empirical studies investigating the relationships between different types of interaction and student learning. For example, Bernard et al. [3] conducted a meta-analysis on the effects of the three types of interactions (i.e., SC, ST and SS) on student performance in online learning. They found that the effects of SS interaction and SC interaction were significantly larger than the effect of ST interaction in terms of student performance.

In this paper, we use this interaction framework to explore how interaction is related to student performance and course completion rates in online courses by analyzing and exploring LMS interaction data.

2.2 Educational Data Mining in Learning Management Systems

A LMS provides a wide range of features to support interactions between students, teachers, and content [9]. Moreover, the LMS typically captures interactions with these features in various formats and at diverse granularity levels. The most widely used methods in EDM studies using LMS data are prediction, clustering, and distillation for human judgment (visualization) [10]. Prior studies have found that usage variables related to SS interaction (i.e., the number of discussion messages posted) and SC interaction (i.e., the number of completed assignments) were significant predictors of student performance [6, 12].

However, prior studies using LMS data analyzed student-level data, rather than looking at the various levels and kinds of interactions between teachers, students, and contents. In this paper, we used course level data as well as individual student level data to provide both macro- and micro-perspectives on interactions between students, teacher, and contents in online learning. In this way, our research complements the existing research base.

3. DATASET AND METHODS

3.1 Dataset

For the present study, data were extracted from the Canvas LMS deployed at a mid-sized public university located in the western U.S. The LMS automatically captures all teacher and student online interactions. Note that an academic support unit at the university extracted and anonymized these data, and Institutional Review Board (IRB) approved using the data for research purposes.

We conducted data preprocessing by transforming raw data into an appropriate shape for analysis. First, we performed data cleaning in the following three steps: 1) selected courses offered between Fall 2014 and Spring 2015; 2) selected only online undergraduate courses; and 3) excluded low enrollment courses (i.e., the number of enrolled students is less than 5). After conducting the data cleaning process, our dataset consisted of 450 courses including 10,576,718 interactions, and anonymized 21,171 student profiles (8,844 distinct student profiles) and 450 teacher profiles (228 distinct teacher profiles).

Table 1 shows the number of courses in our dataset, categorized by STEM vs. non-STEM, size, and course level. 25.8% courses are Science, Technology, Engineering, and Mathematic (STEM) courses. A full range of course sizes is represented and is centered around medium-sized enrollments (i.e., 21-50 students). The largest number of courses is 1000 level (34.7%) and 3000 level (34.9%) courses.

3.2 Data Mining Methods and Features

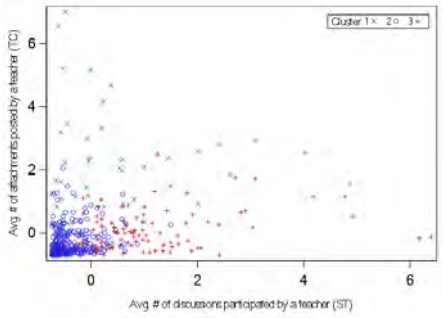
In this study, we used three data mining methods for three tasks – one method for each task: (i) K-means clustering to find groups of courses each of which has similar interaction patterns at a course level; (ii) multiple regression to measure the relationship between each interaction feature/variable and average student final grade and course completion rates at a course level; and (iii) classification algorithms to predict each student's final grade and whether the student will complete a course or not. The first two methods provided a macro perspective focusing on courses, while the last method provided a micro perspective focusing on individual students.

Task 1. We used K-means clustering to identify how online courses were clustered based on interaction patterns. We used the PROC FASTCLUS method in SAS, as missing values were replaced with an adjusted distance using the non-missing values [8]. We used Euclidean distance to measure distance between each node (i.e., a course) and a centroid. To find the optimal K, we examined the agglomeration schedule to determine the optimal number of clusters.

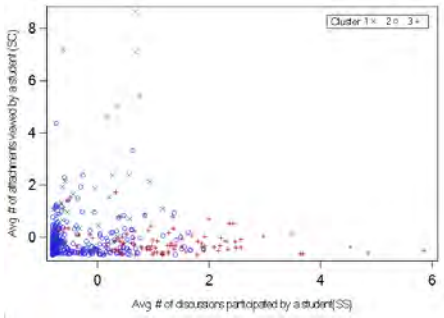
Task 2. We conducted multiple regressions using SAS to test whether each interaction type significantly predicted outcome variables – average final grades and course completion rates.

For Tasks 1 and 2, we grouped Canvas features (variables) into four categories (TC, SC, SS, ST) based on Anderson and Garrison's interaction framework [2]. Table 2 presents four categories associated with the Canvas features, and each feature's mean, standard deviation (SD) and minimum and maximum values obtained from the 450 courses.

Task 3. We applied classification algorithms (i.e., SVM, Random Forest, J48 and AdaBoost) to predict each student's final grade and whether the student will complete a course or not. Effectiveness of classifiers depends on quality of features. For this task, we used 129 features consisting of 52 static features and 77 temporal features as shown in Table 3. These features consisted of not only the main interaction features that we used in the first and second tasks (while they were average values in the first and second tasks, individual student feature values were used in the third task),



(a) ST interaction vs. TC interaction (z-transformed data).



(b) SS interaction vs. SC interaction (z-transformed data).

Figure 1: Scatter plots showing how courses in clusters are distributed differently.

Table 2: Descriptive statistics of 450 courses analyzed by 12 interaction features associated with four categories.

Category	Features	Mean	SD	Min-Max
Teacher-Content	Avg. # of attachments posted by a teacher (tc_atta)	15.97	22.86	0-176
	Avg. # of discussion topics posted by a teacher (tc_disc)	18.55	15.54	0-107
	Avg. # of wiki topics posted by a teacher (tc_wiki)	13.58	13.96	0-74
	Avg. # of quizzes posted by a teacher (tc_quiz)	9.72	9.48	0-56
	Avg. # of assignments posted by a teacher (tc_assi)	15.30	12.97	0-75
Student-Content	Avg. # of attachments viewed by a student (sc_atta)	118.19	174.57	0-1,625
	Avg. # of discussions viewed by a student (sc_disc)	48.05	44.88	0-296
	Avg. # of wiki viewed by a student (sc_wiki)	54.42	51.92	0-387
	Avg. ratio of completed quiz by a student (sc_quiz)	0.88	0.12	0.10-1
	Avg. ratio of completed assignments by a student (sc_assi)	0.78	0.16	0.10-1
Student-Student	Avg. # of discussions participated by a student (ss_disc)	12.21	15.13	0-101
Student-Teacher	Avg. # of discussions participated by a teacher (st_disc)	50.15	68.63	0-489

but also additional features (e.g., the number of views of the grade and announcement pages, course information and temporal features). In particular, temporal features were extracted from a series of daily snapshots of each student's interaction record. Given a course and interaction information of a student who took the course, we represented the student by using the 129 features.

4. EXPERIMENTAL RESULTS

In the previous section, we described our dataset and three data mining methods for conducting three tasks. In this section, we present results of these experiments using each of the methods for each task.

Table 3: 129 Features extracted from each student and each corresponding course.

Static Features	
Features	|Features|
Course level and Department offering the course	2
Total # of views and total # of participation by a student	2
# of views and participation in each of the 24 items by a student	48

Temporal Features	
Features	|Features|
Total # of participated weeks (i.e., we add +1 if a student did participation at least once in a week)	1
Mean and standard deviation of weekly view count and weekly participation count	4
Each week's view count and participation count	36
Accumulated weekly view count and accumulated weekly participation count	36

4.1 Task 1: Clustering Courses and Analyzing Characteristics of Clusters

In Task 1, our research goal was to cluster courses based on interaction patterns and analyze characteristics of the clusters. First, we standardized the interaction features/variables (raw scores) by following the recommendation in the literature [8]. The raw scores were z-transformed to a mean of 0 and standard deviation of 1 for either the course or semester level data.

K-means clustering requires an input K. To make sure we chose an optimal K, we examined the agglomeration schedule. The demarcation point indicated that $K = 3$ would produce the optimal result. Clusters 1, 2 and 3 contained 41, 300 and 109 courses, respectively. The root mean squared standard deviations (RMSSTD) for each cluster were 1.32, 0.71, 0.98 respectively, indicating that the courses in cluster 1 are more widely dispersed than the others.

We further drew two scatter plots to help understand characteristics of the three clusters as shown in Figure 1. Figure 1(a) represents a scatter plot of ST interaction (*st_disc*) vs. TC (*tc_atta*) interaction. Courses in cluster 1 had higher TC interaction than those in the other clusters, whereas courses in cluster 3 had higher ST interaction than the other two clusters. Figure 1(b) shows a scatter plot of SS interaction (*ss_disc*) vs. SC interaction (*sc_atta*). Courses in cluster 1 showed higher student-content interaction than the other two clusters. On the contrary, courses in cluster 3 showed higher SS interaction than the other two clusters.

Table 4: Means and standard deviations of clusters. ∗ indicates the highest value among the three clusters.

Feature	Cluster 1 (n=41) Content-interaction		Cluster 2 (n=300) Low-interaction		Cluster 3 (n=109) Inter-person interaction	
	M	SD	M	SD	M	SD
tc_atta	2.12	1.78	-0.32	0.44	0.09	0.67
tc_disc	0.26	0.96	-0.44	0.59	1.1	1.04
tc_wiki	1.53	1.31	-0.37	0.64	0.43	0.98
tc_quiz	0.68	1.32	-0.05	0.99	-0.12	0.76
tc_assi	0.38	1.23	-0.28	0.77	0.62	1.14
(T-C) mean	0.99*	0.66	-0.29	0.43	0.42	0.55
sc_atta	1.47	2.27	-0.14	0.62	-0.18	0.47
sc_disc	-0.04	0.52	-0.46	0.55	1.22	1.02
sc_wiki	1.8	1.62	-0.23	0.68	-0.07	0.7
sc_quiz	-0.18	1.04	0.02	0.92	0.02	1.19
sc_assi	-0.18	1.15	0.03	1.04	-0.01	0.84
(S-C) mean	0.57*	0.85	-0.16	0.46	0.2	0.42
(S-S)	-0.2	0.59	-0.38	0.58	1.05*	1.22
(S-T)	0.29	1.02	-0.43	0.33	1.07*	1.33
final grades	**2.77**	**0.59**	**3.01**	**0.57**	**3.05***	**0.38**
complet. rates	**84.04**	**12.95**	**86.84**	**12.75**	**88.09***	**9.18**

Next, we examined descriptive statistics for the predictors and outcome variables (final grades and completion rates) for each cluster as shown in Table 4[1]. The results showed that cluster 1, dubbed *"Content-Interaction courses"*, had the highest means for both TC interaction (*M = 0.99, SD = 0.66*) and SC interaction (*M = 0.57, SD = 0.85*). Cluster 2, dubbed *"Low-Interaction courses"*, had the lowest means for all interaction variables. Lastly, cluster 3, dubbed *"Inter-person Interaction"*, had higher means for SS interaction (*M = 1.05, SD = 1.22*) and ST interaction (*M = 1.07, SD = 1.33*). The analysis revealed that courses in each cluster had different course emphases: content interaction in cluster 1, non-interaction in cluster 2, and person interaction in cluster 3.

Then, we compared the three clusters in terms of average student final grades and course completion rates. As shown in Table 4, the cluster 3 had the highest mean in student final grades (*M = 3.05, SD = 0.38*) and course completion rates (*M = 88.09, SD = 9.18*) among the three clusters. The cluster 1 had the lowest mean in student final grades (*M = 2.77, SD = 0.59*) and course completion rates (*M = 84.04, SD = 12.95*). This finding reveals that the positive impact of courses focusing on interactions between participants.

Next, we conducted chi-squared tests to compare STEM and Non-STEM courses in the three clusters. As shown in Table 5, the distribution of the STEM and Non-STEM courses was significantly different across the three clusters, $\chi^2(6, N = 450) = 7.80$, $p < .05$. STEM courses were infrequent overall, but even more scarce in the cluster 3.

Then, we analyzed how many courses in the three clusters had small, medium and large enrollments. Table 6 shows the analytical results. The result of a chi-squared test showed significant differences among the three clusters, $\chi^2(4, N = 450) = 15.31$, $p < .05$. The cluster 1 had the largest proportion of large courses, whereas the cluster 3 had the smallest proportion of large courses. The findings suggest that promoting interaction among participants is rarer in large courses.

[1]The meaning of each feature's acronym is described in Table 2.

Table 5: The number of STEM and Non-STEM courses in three clusters.

Cluster	Non-STEM	STEM	Total
C1	29 (70.7%)	12 (29.3%)	41
C2	21 (71.0%)	87 (29.0%)	300
C3	92 (84.4%)	17 (15.6%)	109
Total	334	116	450

Table 6: The number of small, medium, large courses in three clusters.

Cluster	Small	Medium	Large	Total
C1	13(31.7%)	13(31.7%)	15(36.6%)	41
C2	78(26.0%)	130(43.3%)	92(30.7%)	300
C3	16(14.6%)	67(61.4%)	26(24.0%)	109
Total	107	210	133	450

Lastly, we examined how many courses in the three clusters were at the 1000, 2000, 3000 and 4000 levels. A chi-squared test found no significant differences in the distribution of the course levels among the clusters, $\chi^2(6, N = 450) = 8.79$, $p > .05$.

4.2 Task 2: Prediction Using Multiple Regression Analysis

In task 2, first we conducted a multiple regression analysis to examine the influence of interaction features or feature category listed in Table 2 in predicting average student final grades in each course. Table 7 shows regression results of significant variables. The results indicated that the explanatory variables accounted for a modest 15.8% of the variance ($R^2 = 0.16$, $F(12, 411) = 6.41$, $p < .05$). Several significant and negative predictors were found in teacher-content interaction. In particular, as *tc_disc*, *tc_wiki*, and *tc_assi* increased, final grades tended to decrease. Findings in the student-content interaction category were the opposite. Final grades tended to increase when *sc_quiz* and *sc_assi* increased and the same is true in the student-teacher interaction category.

A second multiple regression analysis was conducted to test the influence of each interaction feature or each feature category on course completion rates. The explained variance was a modest at 15.7%($R^2 = 0.16$, $F(12, 411) = 6.64$). Only a single teacher-content variable *tc_wiki* was negatively significant. Student-content interaction features *sc_quiz* and *sc_assi* were significant and positive again in relation to course completion rates. Taken together, these findings suggest that certain teacher activities related to content were less productive, whereas student activities related to content were more positively productive in both final grades and course completion rates.

Table 7: Multiple regression results (* indicates the feature is significant at the 0.05 level, and the table includes only significant features).

		final grades					completion rates				
Category	**Feature**	***B***	***SE(B)***	β	**t**	**p**	***B***	***SE(B)***	β	**t**	**p**
Intercept		0.000	0.089	0.000	29.600	0.001	0.000	0.089	0.000	29.600	<.0001
Teacher-Content Interaction	tc_disc	-0.006	0.003	-0.177*	-2.240	0.026	-0.078	0.060	-0.059	-0.990	0.324
	tc_wiki	-0.011	0.002	-0.295*	-4.540	0.001	-0.241	0.054	-0.202*	-3.710	0.000
	tc_assi	0.004	0.002	0.106*	1.970	0.050	0.037	0.048	0.033	0.690	0.490
Student-Content Interaction	sc_wiki	0.001	0.001	0.141*	2.140	0.033	0.125	0.015	0.029	1.900	0.058
	sc_quiz	0.003	0.001	0.164*	3.250	0.001	0.284	0.019	0.107*	5.650	<.0001
	sc_assi	0.003	0.001	0.177*	3.530	0.001	0.115	0.019	0.044*	2.290	0.023
Student-Teacher Interaction	st_disc	0.001	0.001	0.160*	2.340	0.020	0.130	0.011	0.022	1.910	0.057

Table 8: Feature Sets

Feature Set	**Features (# of features)**
A	Course level and department offering the course, total # of views and total # of participation (4)
B	feature set *A* + # of views and participation in each of the 24 items by a student (52)
C	feature set *B* + total # of participated weeks (53)
D	feature set *C* + mean and standard deviation of weekly view count and weekly participation count (57)
E	feature set *D* + each week's view count and participation count, and accumulated weekly view count and participation count (129)

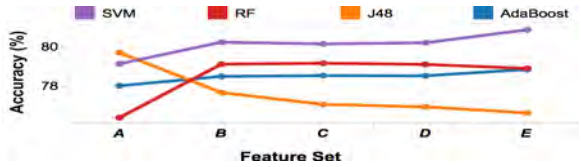
4.3 Task 3: Predicting Individual Student's Final Grade and Course Completion

So far, experiments in Tasks 1 and 2 were conducted at the course levels, providing a macro perspective. Now we turn to building classifiers to predict individual student's final grade and course completion (i.e., whether the student will complete the course or not) by using a data-driven approach, providing a micro perspective, and then evaluating effectiveness of the classifiers. In task 3, predicting a student's final grade means predicting whether the student will belong to a high performance group (i.e., obtaining one of A, A-, B+, B and B-) or a low performance group (i.e., obtaining one of C+, C, C-, D+, D, F and W).

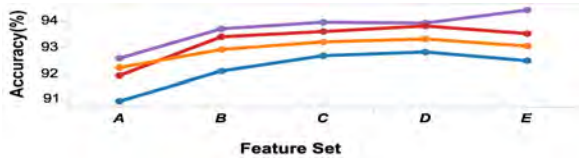
4.3.1 Prediction in 2014 Fall Semester Dataset

In this experiment, we used the 2014 Fall semester dataset consisting of 229 courses with 4,314,425 interactions and anonymized 10,003 student profiles. To build highly accurate classifiers, proposing and using features which have significant distinguishing power is important. To test this, the 129 features listed in Table 3 were sampled to make five feature sets entitled feature sets *A*, *B*, *C*, *D* and *E* as shown in Table 8. As we chose from feature set *A* to *E*, the number of features increased by including the previous features but also additional features. Feature sets *A* and *B* consisted of only static features, while feature sets *C*, *D* and *E* consisted of static features and temporal features.

Since we didn't know apriori which classification algorithm would perform the best, we chose 4 popular classification algorithms – SVM, Random Forest, J48 and AdaBoost. Given the 2014 Fall semester dataset, we did 10-fold cross-validation by dividing the dataset to 10 sub-samples. Each sub-sample



(a) Final grades.



(b) Course completion.

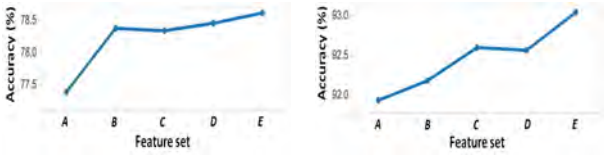
Figure 2: Prediction results of SVM, Random Forest, J48 and AdaBoost based classifiers with five feature sets.

became a test set, the other 9 sub-samples became a training set. We conducted a classification experiment for each of the 10 pairs of training and test sets. Then, we averaged the 10 classification results. We repeated this process for each classification algorithm.

Figure 2 shows prediction results for final grades/performance groups and course completions. SVM based classifier outperformed Random Forest, J48 and AdaBoost based classifiers, achieving 80.95% accuracy, 0.79 F-measure and 0.72 AUC in final grade prediction and 94.41% accuracy, 0.94 F-measure and 0.85 AUC in course completion prediction. As we added more features (changing from feature set ***A*** to ***E***), SVM classifier's accuracy has increased in both predictions. Compared with the *baseline*, which was measured by a percent of the majority class instances and achieved 68% accuracy in final grade prediction and 84% in course completion prediction, our SVM based classifier improved 19% ($= \frac{80.95}{68} - 1$) accuracy in final grades prediction, and 12.4% ($= \frac{94.41}{84} - 1$) accuracy in course completion prediction.

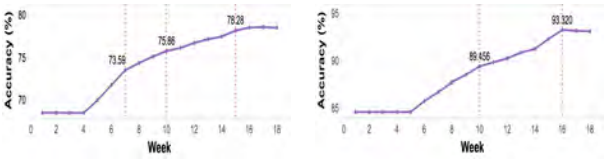
4.3.2 Robustness of Our Prediction Model

In Section 4.3.1, we evaluated effectiveness of our classification approach for both final grades prediction and course completion prediction. Now we are interested in how much the pre-built model is robust when we apply it to data generated in the future (i.e., future semesters). To simulate this scenario, we used the 2014 Fall semester dataset as a



(a) Final grades. (b) Course completion.

Figure 3: Prediction results obtained by applying SVM-based classifiers trained by 2014 Fall dataset to 2015 Spring dataset.



(a) Final grades. (b) Course completion.

Figure 4: Prediction results over time.

training set and the 2015 Spring semester dataset as a test set (consisting of 221 courses with 6,262,293 interactions and anonymized 11,168 student profiles). We built a SVM-based classifier and predicted each student's final grade and course completion in the test set.

Figure 3 shows prediction results as we used feature set *A* to *E*. Again, using all the features (feature set *E*) produced the best results, achieving 78.64% accuracy and 0.682 AUC in final grades prediction and 93.06% accuracy and 0.817 AUC in course completion prediction. Compared with the previous experimental results in Section 4.3.1, there were only small reductions – 2.31% (final grades) and 1.35% (course completion). The experimental results confirmed that our proposed approach is robust and can be applied to future semesters.

4.3.3 Early Prediction

The previous experimental results showed that our approach was effective in predicting final grades and course completion. In practice, it is better to produce prediction earlier so that a tool/system can automatically identify and alert which students are at risk of receiving a low grade or dropping out a course thereby requiring intervention by a teacher. To address this need, we used daily snapshot of data including student profiles, course information and interaction logs, and then simulated the scenario by building a SVM-based classifier in each week. In other words, we built a classifier and evaluated its performance in each week. By doing this, we examined how the classifier's performance changed over time, and when we could achieve a reasonable accuracy.

Figure 4 shows prediction results in the 2014 Fall dataset. In final grades prediction, when we built classifiers in the 7th week, 10th week and 15th week, we achieved 73.59%, 75.86% and 78.28% accuracy, respectively. Similarly, in course completion prediction, we achieved 89.4% and 93.3% accuracy in 10th week and 16th week, respectively. Overall, adding more data improved performance of our classifiers. This study reveals that it is possible to detect students early who have a higher chance of receiving low grades or dropping out a course.

5. CONCLUSIONS

The purpose of this study was to explore relationships between theoretically defined constructs extracted from a Learning Management System and student learning outcomes. Three different tasks employing three different methods were used to explore these relationships. The first two tasks were conducted at the macro-level and thus aligned with a theory-driven approach, whereas the last task at the micro level aligned with a data-driven approach.

Results from the cluster analysis revealed that courses with high inter-person (SS, ST) interaction had higher final grades and completion rates than courses in the other clusters (low-interaction and content-interaction), aligning with results from previous studies [6, 12]. Results also suggested that STEM and large courses tended to exhibit fewer of these productive interactions. The micro-level, data-driven machine learning analysis using prediction with SVM enabled the discovery of at-risk students with high accuracy. It achieved the best performance when all temporal features (complete feature set) were taken into consideration and was robust when predicting future data.

In sum, for this dataset comprised of LMS interactions drawn from online undergraduate courses, the interaction framework was useful for interpreting at both macro and micro levels.

6. REFERENCES

[1] T. Anderson. Modes of interaction in distance education: Recent developments and research questions. *Handbook of distance education*, pages 129–144, 2003.

[2] T. D. Anderson and D. R. Garrison. Learning in a networked world: New roles and responsibilities. 1998.

[3] R. M. Bernard, P. C. Abrami, E. Borokhovski, C. A. Wade, R. M. Tamim, M. A. Surkes, and E. C. Bethel. A meta-analysis of three types of interaction treatments in distance education. *Review of Educational research*, 79(3):1243–1289, 2009.

[4] D. R. Garrison and M. Cleveland-Innes. Facilitating cognitive presence in online learning: Interaction is not enough. *The American Journal of Distance Education*, 19(3), 2005.

[5] D. Laurillard. 8 new technologies, students and the curriculum. *Higher education re-formed*, 2000.

[6] L. P. Macfadyen and S. Dawson. Numbers are not enough. why e-learning analytics failed to inform an institutional strategic plan. *Educational Technology & Society*, 15(3), 2012.

[7] M. G. Moore. Editorial: Three types of interaction. *American Journal of Distance Education*, 3(2):1–7, 1989.

[8] E. Reiss, S. Archer, R. Armacost, Y. Sun, and Y. Fu. Using sas® proc cluster to determine university benchmarking peers. *SESUG, Savannah GA, September*, 2010.

[9] C. Romero, P. G. Espejo, A. Zafra, J. R. Romero, and S. Ventura. Web usage mining for predicting final marks of students that use moodle courses. *Computer Applications in Engineering Education*, 21(1), 2013.

[10] C. Romero, S. Ventura, and E. García. Data mining in course management systems: Moodle case study and tutorial. *Comput. Educ.*, 51(1):368–384, Aug. 2008.

[11] G. Siemens and R. S. d Baker. Learning analytics and educational data mining: towards communication and collaboration. In *Proceedings of the 2nd international conference on learning analytics and knowledge*, 2012.

[12] T. Yu and I.-H. Jo. Educational technology approach toward learning analytics: Relationship between student online behavior and learning performance in higher education. In *Proceedings of the Fourth International Conference on Learning Analytics and Knowledge*, 2014.

A Comparison of Automatic Teaching Strategies for Heterogeneous Student Populations

Benjamin Clement
Inria, France
benjamin.clement@inria.fr

Pierre-Yves Oudeyer
Inria, France
pierre-yves.oudeyer@inria.fr

Manuel Lopes
Inria, France
INESC-ID, Instituto Superior Técnico, Portugal
manuel.lopes@inria.fr

ABSTRACT

Online planning of good teaching sequences has the potential to provide a truly personalized teaching experience with a huge impact on the motivation and learning of students. In this work we compare two main approaches to achieve such a goal, POMDPs that can find an optimal long-term path, and Multi-armed bandits that optimize policies locally and greedily but that are computationally more efficient while requiring a simpler learner model. Even with the availability of data from several tutoring systems, it is never possible to have a highly accurate student model or one that is tuned for each particular student. We study what is the impact of the quality of the student model on the final results obtained with the two algorithms. Our hypothesis is that the higher flexibility of multi-armed bandits in terms of the complexity and precision of the student model will compensate for the lack of longer term planning featured in POMDPs. We present several simulated results showing the limits and robustness of each approach and a comparison of heterogeneous populations of students.

1. INTRODUCTION

The current advances and ubiquity of learning and teaching technologies have the potential to improve education accessibility and personalization. Intelligent Tutoring Systems (ITS) have been proposed to make education more accessible, more effective, and as a way to provide useful objective metrics on learning [1].

A major aspect of personalized education is to be able to identify the current level of students and how to address particular difficulties in the student learning process. The goal is to be able to choose online the activity that better addresses the challenges being encountered by each particular student. Even two students with the same knowledge will require different activities to progress further due to their previous experience, cognitive skills or preferences. This is a difficult challenge because as ITS are encountering the students for the first time, it is difficult to know what is the impact of each activity on their progress. A commonly used method is to exploit a population-wide model on how students learn and assume that they are all similar. The personalization in such an approach is limited to adapting to student's knowledge levels but assumes that the impact of each exercise is the same for all students with the same knowledge levels.

Different methods have been proposed to handle this problem. One popular and well-known method is the Partially Observable Markov Decision Process (POMDP) framework which has been proposed in different ways to select the optimal activities to propose to a learner [13]. This framework can find the optimal teaching trajectories for a given teaching scenario model if an accurate student model is provided which is not always possible in practice. The main drawback is the high computational complexity and as a consequence, only the simplest cases can be solved exactly. Another method explored recently to select optimized activities is to use the Multi-Arm Bandit (MAB) framework to personalize sequences of pedagogical activities [6]. These methods optimize learning in the short term (rather than in the long-term) and rely on much simpler student models while being computationally very efficient.

In this paper, we compare the POMDP framework and the MAB framework (specifically the algorithm ZPDES already evaluated in real classrooms [6]). We first introduce a student model used to compare the different algorithms. We then propose ways to model the heterogeneity in classrooms by considering that different students will have not only different learning parameters but also that they might have different dependencies between the different knowledge components (KCs). Our experiments will evaluate how well a MAB can approach the optimal solution of a POMDP, and how the different algorithms behave when encountering a heterogeneous group of student.

2. RELATED WORK

In this work we are interested in the impact of the quality of the student models on the quality of the sequences of activities chosen by online algorithms.

There are several approaches to automatically choose exercises based on the current knowledge level of students. We are here particularly interested in optimization methods that rely on minimal prior assumptions about the students or the knowledge domain.

One option already explored is the use of a partial-observable Markov decision process (POMDP) [13], [14]. PODMPs offer an appealing theoretical framework that guarantees an optimal long-term solution for a planning problem. However, in general, as the computational complexity is high, it is practically impossible to find an exact solution to the problem. Some approximate solutions in the domain of ITS have considered the use of aggregations of states instead of tracking the full knowledge components. Another drawback is that POMDPs require a precise student model for which the policy is optimized. If the real student encountered deviates from this model, then the optimality properties are lost.

A more recent approach is to use the Multi-Arm Bandit (MAB) framework to manage pedagogical activities [6]. MABs have the advantage of being extremely computationally efficient and rely on very weak student models. The main drawback is that there is no long-term planning of the best sequence of activities relying on an exploration-exploitation tradeoff to find the best path. Aware of such problem, authors of one such algorithm considered that standard MAB needs to be complemented with a weakly specified knowledge graph to provide a long-term view on the optimization [6].

As noted, before optimizing the sequence of exercises it is important to have some knowledge about the impact of a given exercise in the learning of the KCs, and also to be able to track what each student already masters. A large part of ITS research has been on the modeling aspects of the cognitive and student models. A seminal work on this topic was the *Knowledge Tracing* framework [7] which builds a detailed cognitive model of the student, of its learning processes by considering a set of independent KCs, the probability of learning them and the probability of correct or wrong answer in exercises that relies on those KCs. More recent methods extend this framework to a bayesian probabilistic approach [12, 15] improving the performance and understanding of those methods. Recent methods have started to consider how to learn such models, and variants of it, allowing to simultaneously discover the relation between activities and KC, e.g. [8, 2, 5, 9].

As discussed these methods require an accurate knowledge of how students learn and require to track their mastery of each KC. For this, it is necessary to learn the constraints between different KC, exercises and KC. Given students' particularities, it is impossible for a teacher to understand all the difficulties and strengths of individual students and provide an accurate student model manually. Even with the recent advances on model learning, there are several challenges in identifying parameters that best describe each individual student. These models have many parameters, and identifying all such parameters for a single student is a very hard problem due to the lack of data, often making the problem intractable. In most cases it is even impossible to identify some of the parameters [3, 4]. In the general case, it results in inaccurate models that cannot be exploited for individualized learning. Another problem is that these planning methods are for a population of students and not for a particular student and this has already been proven to be suboptimal [11].

3. STUDENT MODELS

3.1 Student model

In this section, we will present the student model we will use, also called learner model in literature. We want a generative model that can simultaneously be used to predict students behaviour, model their knowledge acquisition and track their mastery level. For this, we built a student model, shown in Fig.1 similar to the Knowledge Tracing framework [10] and its variants. Similarly to [9], we include extra features in our model. We are particular interested in more realistic cases where each KC might depend on other KCs. In most cases it is assumed that each exercise just depends on one KC and that they are independent, this is not realistic most of the time, and such dependencies have a strong impact on the learning sequences generated by the different algorithms.

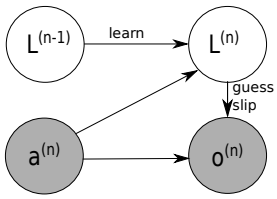


Figure 1: Graphical model of the Student model, with $L^{(n)}$ the hidden state of the student at step n, $a^{(n)}$ activity proposed, and $o^{(n)}$ the result obtained by the student.

We consider a situation where a student has a set of m KCs K_i to learn. A student's state at step n is represented by the state of each KC, $L^{(n)} = K_1^{(n)}, \ldots, K_m^{(n)}$, the global model is described on figure Fig.1. Each KC is defined by his state, mastered ($K_i = 1$) or not mastered ($K_i = 0$). For each KC, there is an initial probability of mastering it $p(K_i^{(0)} = 1)$ which is always null in our experiments to make students learn all the KCs through activities. The emission probabilities are defined by the guess probability, i.e performing correctly without mastering the skill, and the slip probability, i.e performing incorrectly despite mastering the knowledge. Theses probabilities are constant. Finally $p(K_i^{(n)} = 1|L^{(n-1)}, a^{(n)})$ defines the probability of transition from not mastered to mastered K_i while doing activity a at step n and depending of the constraints between KCs and their states. An activity can be represented as a vector $a = \alpha_1, \ldots, \alpha_m$ where $\alpha_i = 1$ if the activity allows to acquire K_i, $\alpha_i = 0$ else. The transition probability to learn a given KC K_i at step n is given by the following formula:

$$p(K_i^{(n)} = 1|L^{(n-1)}, a^{(n)}) = \alpha_i(\beta_{i,i} + \sum_{j \neq i}^{m} \beta_{i,j} K_j^{(n-1)}) \quad (1)$$

Where $\beta_{i,i}$ represent the probability to learn K_i without considering other KCs and $\beta_{i,j}$ represent the impact of the KC K_j on the probability to learn K_i. If a given KC does not need other KCs to be learned, the term $\sum_{j \neq i}^{m} \beta_{i,j} K_j$ is null.

For more simplicity, in our experiments, an activity a can provide an opportunity to acquire only one KC which induces an isomorphism between the knowledge space and the activity space.

3.2 Models of populations

The previous model can be used to describe a single student or an average model of a population. Our goal is to understand the impact that the diversity of students has when the given sequence is optimized considering the same parameters for all students. We will achieve such goal by considering a canonical model and then make two types of disruptions: i) change the probabilities between the variables; ii) change the knowledge graph.

The first way is to disrupt the parameters in the model, i.e. the probability of transition, guess, and slip. To do that, we consider that each parameter is sampled from a gaussian distribution. We can change the variance to increase the heterogeneity of the population. With a variance null, all the population has the same parameters. The second way is to change the knowledge graph that changes the dependencies between the different knowledge. This type of disruption can be small like adding or removing a dependency, or it can be as critical as rearranging completely the organization of the knowledge dependencies. These two types of disruption are combined in our experiments.

$$LM_0 : \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_6$$

$$LM_1 : \left.\begin{array}{l}\xrightarrow{0.2} K_1 \\ \xrightarrow{0.2} K_2\end{array}\right\} \xrightarrow[0.1]{0.1} K_3 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_6$$

$$LM_2 : \begin{array}{l}\xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_6 \\ \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_5\end{array}$$

$$LM_3 : \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_6$$

$$LM_4 : \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_6$$

Figure 2: Knowledge graphs used in the simulations. LM_0 is the nominal knowledge graph, with LM_1 and LM_2 introducing small disruptions in the pre-requirements between KCs. LM_3 and LM_4 represent more critical disruptions that change the overall order of KCs.

We used multiple knowledge graphs, shown in Fig.2. The arrows represent the dependencies between KCs. For example, LM_0 represents a graph where the constraints between the different KC are ordered in a linear way. Here, $\beta_{1,1} = \beta_{2,1} = \beta_{3,2} = \beta_{4,3} = \beta_{5,4} = \beta_{6,5} = 0.2$ and all the others values of $\beta_{i,j}$ are null. We then created several different transformations and variants to model different needs of the students in terms of the order of the different KC.

LM_1 and LM_2 follow approximately the same overall sequence of KC, but considering two initial branches for the different KC. LM_1 considers that KC_1 and KC_2 are independent and any of them allows to learn KC_3. In these knowledge graphs, we can expect that optimizing for one will also work for the other as the overall sequence of KC is respected, even if the strategy is no longer optimal. We also created more critical disruptions in the knowledge graph. LM_3 and LM_4 present an inversion between two KCs. For LM_3, KC_4 and KC_5 are inverted, what radically change the overall sequence of KCs. For LM_4, it is K_3 and K_4 that are inverted.

4. OPTIMIZING LEARNING POLICIES

4.1 Partially Observed Markov Decision Process (POMDP)

POMDP is a markovian decision process where the state is hidden and can only be inferred indirectly from the observations. A POMDP consists of a tuple $\langle S, A, Z, T, R, O, \gamma \rangle$ with S the state space, A the action space and Z the observation space. T is the transition model, it gives the probabilities $p(s'|s, a)$ of transitioning from state s to state s' with the action a. O is the observation model, it gives the probabilities $p(z|s, a)$ of having the observation z when action a is made in state s. R the cost model, it specifies the cost $r(s, a)$ of choosing action a in state s, and the discount factor γ gives the relation between immediate costs and delayed costs. With all these components, the solution of a POMDP is a policy that optimizes total discounted future reward.

This framework has been already used in the context of ITS [13]. The learner's mastery is the hidden state s, learning is the transition between states, the probabilities that the learner gives a good answer are given by the observation model of the observation {correct, incorrect}. We use Perseus [14] as solver to find the optimal policy for our POMDP problem.

4.2 Zone of Proximal Development and Empirical Success (ZPDES)

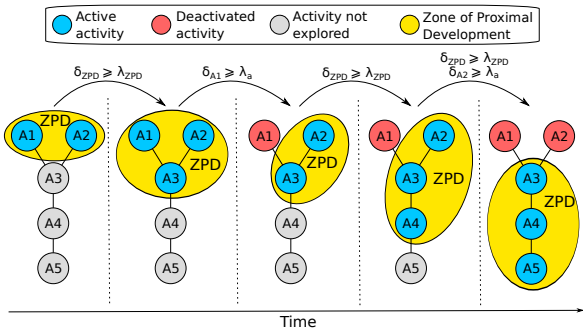


Figure 3: ZPDES exploration of an activity graph, with δ_{ZDP} the success rate over all active activities, λ_{ZPD} the threshold to expand the ZPD, δ_{Ax} the success rate for the activity Ax, and λ_a the threshold to reach to deactivate an activity.

Here we present the recently introduced algorithm Zone of Proximal Development and Empirical Success (ZPDES) that is based on multi-armed bandits [6]. The idea of the algorithm is presented in Fig.3 and summarized in Alg.1. The algorithm follows an activity graph but goes through it in a stochastic way. ZPDES is initialized with a certain number of activities defined as starting activities. At each point in time, ZPDES has a set of activities, called the zone of proximal development, that can be proposed to the student which is adapted depending on student result. In the experiments presented here, we make small changes in the activation/deactivation mechanism of the original algorithm. When the recent student success rate over all active activi-

ties δ_{ZPD} reaches a value λ_{ZPD}, the graph is expanded to explore another activity and when the recent success rate for a particular activity δ_{a_i} is higher than a threshold λ_a, this activity can be removed from the active list. This two threshold allow to partially configure the exploration behaviour of the algorithm. Inside the set of active activities, ZPDES proposes exercises proportionally to the recent learning progress obtained by that activity. The activity graph following the same structure than the knowledge graph, we can directly configure ZPDES with the same knowledge graph used to configure POMDP.

Algorithm 1 ZPDES algorithm

Require: Set of n_a activities A
Require: ζ rate of exploration
Require: distribution for parameter exploration ξ_u
Initialize of quality w_a uniformly
while *learning* **do**
  Initialize ZPD
  {Generate exercise:}
  for $a \in ZPD$ **do**
    $\tilde{w}_a = \frac{w_a}{\sum_j w_j}$
    $p_a = \tilde{w}_a(1-\zeta) + \zeta\xi_u$
    Sample a proportional to p_a
  end for
  Propose activity a
  Get student answer C_t and compute reward:
  $r = \sum_{k=t-d/2}^{t} \frac{C_k}{d/2} - \sum_{k=t-d}^{t-d/2} \frac{C_k}{d-d/2}$
  $w_a \leftarrow \beta w_a + \eta r$ {Update quality of activity}
  Update ZPD based on activity graph and success rates
end while

5. EXPERIMENTS

The goal of our experiments is to compare the impact of the knowledge about the students on the online algorithms for choosing exercises, namely POMDP and ZPDES. We will proceed to change the heterogeneity of the student populations and see how much disruption each algorithm is able to adapt. Our comparative measure of performance is the average skill level overall knowledge and over time, for all the students in the population.

We will compare the results obtained with two algorithms: POMDP and ZPDES. Each algorithm will have different variants based on the knowledge included on each of them. POMDP relies on a knowledge graph and the parameters of such graph. Each variant of $POMDP_x$ is characterized by a specific student model used to find the optimal policy. ZPDES has as information the knowledge graph, and some parameters describing how to traverse this graph, no particular assumption is made about the probabilities of knowledge acquisition. $ZPDES_x^H$ is a variant of ZPDES with the corresponding graph x and using the parameters that were used in an other experiment in a real world situation [6] mostly hand-tuned with the help of a pedagogical expert. $ZPDES_x^*$ will also use the graph x but the parameters to traverse the graph are optimized for that particular graph using a greed search. During the optimization, we saw that the majority of parameters present average results and only extreme parameters gave critical results.

Single model results. The first experiment will do a sanity check to evaluate each algorithm in conditions where each student is the same in the population and each algorithm is configured for this model of student. We expect POMDP to have the best results and we want to see how far ZPDES will be from the optimal solution. A Random strategy which selects one activity randomly among all possible is also presented in this first experiment to see the gain of the algorithms.

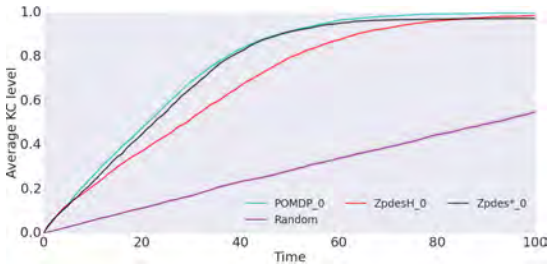


Figure 4: Evolution of the average skill level for 600 students modeled with LM 0 which activity are managed by POMDP, ZPDES*, ZPDES^H configured for LM 0. Shaded area represents the standard error of the mean.

Fig.4 shows the comparison of POMDP, ZPDES*, ZPDES^H and Random with a population of 600 students modelled with the knowledge graphs LM 0. We can see POMDP is the best for all the models, closely followed by ZPDES*. ZPDES^H give a slower learning than the two others. Unsurprisingly, for one particular model, POMDP has the best performance. The optimized ZPDES is very close in performance to POMDP. The results are similar for models 1, 2, 3 and 4, the curves are not presented here for space reason. We can thus verify that the combination of knowledge graphs and the activity exploration rules provides a space of policies that is close to the optimal POMDP one. ZPDES^H present the slowest population learning among the algorithms but as its configuration was not optimized for any particular model we can expect such result.

These results show that the algorithms behave as expected and that ZPDES has the potential to be close to the optimal POMDP solution.

Multi model results. We will now present the main results of this work with the comparison between POMDP, ZPDES* and ZPDES^H when confronted with heterogeneous populations of students. The protocol of the experiments is as follows. First we provide each algorithm with the information about a specific population of students and then we test the capability of the algorithms to address a different and diverse population of students. As described earlier, each algorithm is given information about a particular student model x, POMDP$_x$ receives the graph and the student model parameters, ZPDES$_x^*$ receives the graph and exploration parameters optimized for that same graph, ZPDES$_x^H$ receives the graph and standard parameters for the graph exploration. We test different versions of each algorithm with a population composed of students following 3 differ-

Table 1: Performance position of each algorithm configuration for each setup. The rank of each algorithm configuration, and the average rank of each algorithm is presented for steps 50 and 200.

Students 0,1,2 / Alg config 0,1,2

	Rank t 50		Rank t 200	
Algorithm	Per conf	Average	Per conf	Average
$POMDP_0$	1		1	
$POMDP_1$	3	1	2	2
$POMDP_2$	4		3	
$ZPDES_0^H$	3		1	
$ZPDES_1^H$	3	3	1	1
$ZPDES_2^H$	6		3	
$ZPDES_0^*$	2		1	
$ZPDES_1^*$	3	2	2	2
$ZPDES_2^*$	5		3	

Students 0,3,4 / Alg config 0,3,4

	Rank t 50		Rank t 200	
Algorithm	Per conf	Average	Per conf	Average
$POMDP_0$	1		2	
$POMDP_3$	2	1	3	2
$POMDP_4$	4		5	
$ZPDES_0^H$	2		1	
$ZPDES_3^H$	3	2	2	1
$ZPDES_4^H$	4		3	
$ZPDES_0^*$	2		2	
$ZPDES_3^*$	3	2	4	2
$ZPDES_4^*$	4		4	

Students 2,3,4 / Alg config 0,1

	Rank t 50		Rank t 200	
Algorithm	Per conf	Average	Per conf	Average
$POMDP_0$	1	1	3	2
$POMDP_1$	4		6	
$ZPDES_0^H$	2	1	1	1
$ZPDES_1^H$	3		2	
$ZPDES_0^*$	2	1	4	2
$ZPDES_1^*$	3		5	

ent knowledge graphs. The probabilistic parameters of the student models in the population follow a gaussian distribution. There is 200 students per graphs for a total of 600 students.

On figure 5 we can see the evolution of the average mastery level for all KCs. The table 1 presents the ranking of each version of the algorithms and the average ranking of each algorithm at step 50 and 200 according to the curves comparison for each setup $LM_{0,1,2}$, $LM_{0,3,4}$, and $LM_{2,3,4}$. The table 2 presents the statistical significance tests at step 50 and 200 for each setup and what is the best methods if the results are statistically significant.

By comparing the different p-values, we can see that the differences between POMDP and ZPDES* are never significant, but it's not the case for ZPDES^H. For the models $LM_{0,1,2}$, at step 50, ZPDES^H drops behind the two others, but it catches up rapidly with the two others and present the same results at step 200. So for models which are close to each other, the 3 algorithms present almost the same result.

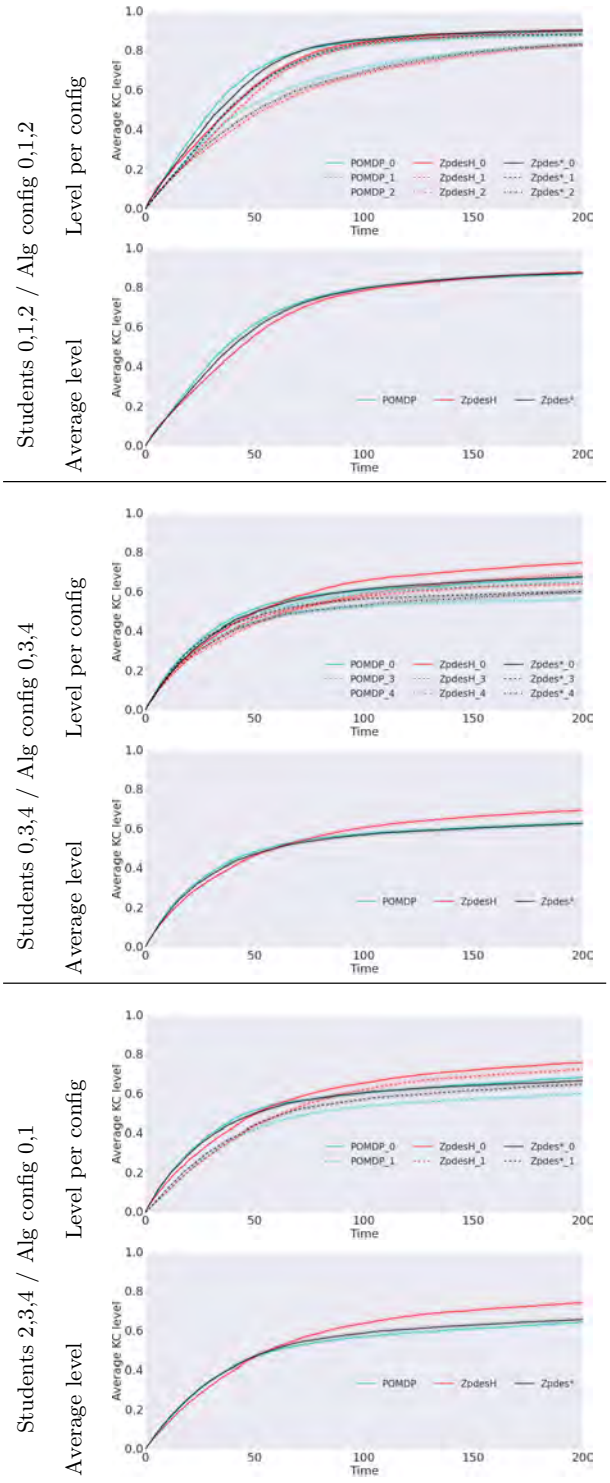


Figure 5: Evolution of the average skill level for 600 students with POMDP, ZPDES*, ZPDES^H. For each curve, the number attached to the algorithm's name indicate what knowledge graph has been used to configure the algorithm. Each curve shows the average KC level of the student population over time for each algorithm configuration. In general ZPDES have better results than POMDP. Shaded area represents the standard error of the mean.

Table 2: ANOVA p-values for each setup to verify if the differences in the KC level distribution according to each algorithm are statistically significant with the best algorithms in parenthesis when it is significant. We note P for POMDP and Z for ZPDES

	P/Z*		P/Z^H		Z*/Z^H	
LM	t 50	t 200	t 50	t 200	t 50	t 200
0,1,2	.075	.95	$\mathbf{10^{-6}}$ **(P)**	.82	**.003** **(Z*)**	.87
0,3,4	.24	.90	.17	$\mathbf{10^{-5}}$ **(Z^H)**	.89	$\mathbf{10^{-4}}$ **(Z^H)**
2,3,4	.31	.30	.18	$\mathbf{10^{-5}}$ **(Z^H)**	.77	$\mathbf{10^{-7}}$ **(Z^H)**

For the models $LM_{0,3,4}$, observations are different. At step 50, all the algorithms seem to have approximately the same performance, even if ZPDES^H seems a bit behind but it's not significant (p-values at 0.17 and 0.89). But with time, it takes the lead and achieves the best performance at 200 steps. So when there are two models critically different from another, ZPDES^H presents the best results. For the last case, the population is constituted of students following $LM_{2,3,4}$ models, and the algorithms are configured for models $LM_{0,1}$. As for the previous case there is no differences at step 50 but ZPDES^H presents the best results at step 200.

ZPDES^H provides the best result because its exploration parameters were not optimized for any particular knowledge graph, giving it higher adaptability and less constrains in the exploration. For a particular type of student model it will present worse performance than POMDP or ZPDES*, but for a heterogeneous population, ZPDES^H, being more adaptable, has the best performance.

6. CONCLUSION

In this work we considered student models where the knowledge components can have constraints among each other, allowing to model some kind of pre-requisites. Under different student models we can find an optimal teaching sequence using POMDP. Another alternative is the use of the recently proposed method ZPDES that is computationally more efficient but without optimality guarantees. Our goal was to test how robust each of these methods is in relation with ill-estimated parameters of the models, or even wrongly estimated relations between KCs. This corresponds to the more realistic case of heterogeneous classes of students.

We showed that for the trivial situation where the students are perfectly modeled with the student model, ZPDES can achieve the same performance as the POMDP. For heterogeneous populations again ZPDES can achieve solutions similar to POMDP. The best algorithm was using ZPDES that uses parameters that are not optimized for no population in particular. By having more flexibility in the exploration it becomes more robust to changes in the population.

We conclude that multi-armed bandits, when combined with an activity graph, are a best choice in comparison with POMDPs due to its computational efficiency and reliance on simpler student models.

The code to generate the graphics and the results is available at: **github.com/flowersteam/kidlearn/tree/edm2016**, follow the README.

7. REFERENCES

[1] John R Anderson, Albert T Corbett, Kenneth R Koedinger, and Ray Pelletier. Cognitive tutors: Lessons learned. *The journal of the learning sciences*, 4(2):167–207, 1995.

[2] Ryan SJ Baker, Albert T Corbett, and Vincent Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *Intelligent Tutoring Systems*, pages 406–415, 2008.

[3] Joseph E Beck and Kai-min Chang. Identifiability: A fundamental problem of student modeling. In *User Modeling 2007*. 2007.

[4] Joseph E Beck and Xiaolu Xiong. Limits to accuracy: How well can we do at student modeling? In *Educational Data Mining*, 2013.

[5] Hao Cen, Kenneth Koedinger, and Brian Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*, 2006.

[6] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, and Manuel Lopes. Multi-Armed Bandits for Intelligent Tutoring Systems. *Journal of Educational Data Mining (JEDM)*, 7(2):20–48, June 2015.

[7] A.T. Corbett and J.R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[8] José P González-Brenes and Jack Mostow. Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. In *EDM*, pages 49–56, 2012.

[9] JP González-Brenes, Yun Huang, and Peter Brusilovsky. General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. In *Inter. Conf. on Educational Data Mining*, 2014.

[10] K.R. Koedinger, J.R. Anderson, W.H. Hadley, M.A. Mark, et al. Intelligent tutoring goes to school in the big city. *Inter. Journal of Artificial Intelligence in Education (IJAIED)*, 8:30–43, 1997.

[11] J.I. Lee and E. Brunskill. The impact on individualizing student models on necessary practice opportunities. In *Inter. Conf. on Educational Data Mining (EDM)*, 2012.

[12] Kai min Chang, Joseph Beck, Jack Mostow, and Albert Corbett. A bayes net toolkit for student modeling in intelligent tutoring systems. In *Intelligent Tutoring Systems*, 2006.

[13] A. Rafferty, E. Brunskill, T. Griffiths, and P. Shafto. Faster teaching by pomdp planning. In *Artificial Intelligence in Education*, pages 280–287, 2011.

[14] Matthijs T. J. Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

[15] Michael Villano. Probabilistic student models: Bayesian belief networks and knowledge space theory. In *Intelligent Tutoring Systems (ITS'92)*, 1992.

Automatic Assessment of Constructed Response Data in a Chemistry Tutor

Scott Crossley
Kristopher Kyle
Georgia State University
Atlanta, GA 30303
scrossley@gsu.edu
kkyle@student.gsu.edu

Jodi Davenport
WestEd
San Francisco, CA 94107
jdavenp@wested.org

Danielle S. McNamara
Arizona State Univ.
Tempe, AZ, 85287
dsmcnama@asu.edu

ABSTRACT
This study introduces the Constructed Response Analysis Tool (CRAT), a freely available tool to automatically assess student responses in online tutoring systems. The study tests CRAT on a dataset of chemistry responses collected in the ChemVLab+. The findings indicate that CRAT can differentiate and classify student responses based on semantic overlap with student input and indices related to word frequency, text content, and lexical sophistication. Overall, the findings suggest that more accurate student responses show greater overlap with the content learned, include more academic function words, contain greater content that is descriptive, and includes more specific and familiar words.

Keywords
Natural language processing, on-line tutors, constructed response scoring

1. INTRODUCTION
For science education to be more effective, students should move beyond memorizing facts and procedures and toward gaining deeper conceptual understanding that allows them to both apply scientific knowledge to explain new phenomena and to design investigations. The Next Generation Science Standards [1], offer a new vision of science instruction that integrates science practices, disciplinary core ideas, and cross cutting concepts, such as scale, energy, and patterns that unify different fields. However, assessing learning of these interconnected strands is challenging using traditional, multiple-choice items. Constructed responses, as well as more novel types of assessments provide students with important opportunities to demonstrate reasoning, explanation, and inquiry skills and are thus an important educational tool [2].

One problem with constructed responses are associated scoring costs [3]. A possible solution to these costs can be found in automated scoring tools that can reduce the need for human scoring and potentially increase scoring consistency [4]. In this study, we introduce a freely available natural language processing (NLP) tool called the Constructed Response Analysis Tool (CRAT) that can automatically score constructed responses in domain specific learning environments. We conduct a pilot study that tests the efficacy of CRAT to score student responses to a domain specific question in an on-line chemistry tutoring system by comparing scoring models developed by CRAT to human ratings of constructed responses.

1.1 Assessing student understanding
Simulations and games provide rich environments for students to learn science and demonstrate their understanding of scientific principles [5]. Such games and simulations can be included in online systems that allow for just-in-time feedback. The dynamic feedback found in online systems affords students the opportunity to confront misconceptions and provides information about areas of struggle or mastery that teachers can use as formative assessments that influence instructional decision making. However, the utility of feedback depends on the ability of an online system to provide an accurate diagnosis of student understanding. Though multiple choice and student behaviors in simulation environments may be readily scored using constraint-based model tutors [6], interpreting and accurately scoring constructed responses in science education has proven much more challenging [2]. These challenges have led researchers to develop content-based automated scoring systems that demonstrate medium to high agreement with human scores. These systems show promise for a number of domains (e.g., math, reading, psychology, biology) and a number of student levels (i.e., middle school, high school, college) [7, 8. 9].

1.2 Current Study
The goal of this study is to introduce CRAT and examine its potential to automatically assign accuracy scores to student constructed responses from an on-line tutor. Constructed responses were collected in the ChemVLab+ tutoring system (chemvlab.org) and scored by expert raters. We used the Constructed Response Analysis Tool (CRAT) to calculate linguistic features related to text content, text summarization, and lexical sophistication and used these linguistic features to predict the human scores.

2. METHOD

2.1 ChemVLab+
The ChemVLab+ is an on-line tutoring system that provides students with opportunities to apply chemistry knowledge to meaningful contexts and to receive immediate, individualized tutoring. Of interest in the current study are the four stoichiometry activities contained within ChemVLab+. The activities engage students in a variety of problem-solving tasks using interactive simulations including a virtual chemistry lab. At the end of each activity, students respond to one to three open-ended questions (i.e., constructed responses) designed to evaluate their ability to synthesize the information they had learned. The four stoichiometry activities included a total of 10 questions.

2.2 Participants

A total of 1392 high school chemistry students from the classes of thirteen teachers in the California bay area used the Stoichiometry module. Students used the online activities as part of their normal coursework.

2.3 Human Scores of Constructed Responses

All constructed responses were coded by two independent raters familiar with the chemistry content. Coders used an annotated rubric that described criteria for each score and provided examples of responses receiving those scores. Reliability of scoring varied across the questions, and interrater reliability ranged from Cohen's $\kappa = 0.55$ to .92. Each question had three possible scores, except for the two lowest reliability questions, (items 1 and 2.1), which had four possible scores. When the highest two scores in these questions were collapsed, interrater reliability increased from 0.56 to 0.68 for item 1 and from 0.59 to 0.69 for item 2.2.

2.4 Selection of Constructed Responses

We selected student constructed responses from question 1 in the stoichiometry lab to test CRAT. The question had the greatest number of student answers (n = 1374). The question asked students to explain the relationship between the amount of sugar, the volume of the drink, and concentration of the sports drink.

2.5 CRAT

CRAT is an easy to use constructed response analysis engine that calculates indices related to a) the linguistic and semantic similarities between a source text and a constructed response, b) the linguistic sophistication of a constructed response, and c) text properties (e.g., length and syntactic categories). It is freely available, cross-platform, and is accessed via a graphic user interface (GUI). The similarity indices include lexical similarity calculated using key word overlap, synonym overlap, and latent semantic analysis (LSA) similarity [10] and phrasal similarity calculated using key bigram and trigram overlap and key part of speech sensitive slot-grams (e.g., a trigram with an open slot such as *into the* ____). The constructed response sophistication indices include psycholinguistic word information indices (e.g., concreteness and familiarity [11, 12]), lexical frequency and range (words that occur in a wider range of texts) indices based on the British National corpus (BNC [13]) and the Corpus of Contemporary American English (COCA [14]), and syntactic categories (e.g., number of adjectives and nouns). For COCA, CRAT reports on frequency and range indices for a number of different genres including academic, newspaper, and fiction genres. Selected index features are outlined below. See http://www.soletlab.com to download the tool and to access the complete list of indices.

2.5.1 Function and content word only indices

CRAT indices generally consider all words in a text. CRAT also includes index variants that include only the content words (e.g., nouns, verbs, adjectives, adverbs) and only the function words (e.g., determiners, prepositions, etc.). Content word indices and function word indices are designed to provide more fine-grained analyses, and have been shown to be more predictive, in some cases, than when all words are considered in an index [15].

2.5.2 Text and sentence minimum indices

CRAT indices generally comprise the average score for all instances of a feature across an entire text. Additionally, CRAT calculates index variants that comprise average minimum scores for each sentence in a text in order to assess smaller texts that may be a single sentence in length.

2.5.3 Key word exclusion indices

In addition to the index variants outlined above, constructed response sophistication indices include variants that exclude words that occur more frequently in the source text than would be expected (i.e., words that are "key"). The key word exclusion index variants were included to minimize interference from sophisticated language in the source text on the constructed response produced.

2.5.4 Latent Semantic Analysis Weighting

One variable that can affect LSA similarity scores is the weighting scheme employed. CRAT includes LSA variants calculated from the TASA corpus using normalized weighting, rare words dominated weighting, and frequent words dominated weighting. Normalized weighting considers all words in a reference corpus equally. Rare words dominated weighting assign higher scores to words that occur infrequently in the reference corpus. Frequent words dominated weighting assigns higher scores to words that frequently occur in the reference corpus [16].

2.6 Summary Input

CRAT is a domain specific tool and uses system input (i.e., source texts) to develop knowledge spaces for the domain of interest. The source texts used to develop knowledge spaces can be textbooks, lecture notes, presentations, or any type of text that generalizes expected knowledge on the part of the student. For this analysis, we used the hints provided to the students during specific activities within the ChemVLab+ system. These hints provide an overview of the input the student received and are designed to provide informational hints to students if they are unable to generate the information individually. The hints available to students in question 1 of the stoichiometry lab comprised over 5,000 words and focused specifically on the relationship between sugar, volume, and concentration in a sports drink.

2.7 Statistical Analysis

The indices reported by CRAT that yielded non-normal distributions were removed. A multivariate analysis of variance (MANOVA) was conducted to examine which indices reported differences between the three levels of scores for each student response (incomplete or incorrect, partially correct, and correct responses). The MANOVA was followed by stepwise discriminant function analysis (DFA) using the selected normally distributed indices from CRAT that demonstrated significant differences between responses that were incorrect or incomplete, partially correct, and correct and did not exhibit multicollinearity ($r > .90$) with other CRAT indices. In the case of multicollinearity between indices, the index demonstrating the largest effect size was retained in the analysis. The DFA was used to develop an algorithm to predict group membership through a discriminant function co-efficient. A DFA model was first developed for the entire corpus of constructed responses. This model was then used to predict group membership of the constructed responses using leave-one-out-cross-validation (LOOCV) in order to ensure that the model was stable across the dataset.

3. RESULTS

3.1 MANOVA

A MANOVA was conducted using the NLP indices calculated by CRAT as the dependent variables and the human scores of the student responses as the independent variables. Of the 759 indices

Table 1: Descriptive statistics and MANOVA results for CRAT variables

Index	Incomplete/incorrect Mean (SD)	Partially correct Mean (SD)	Correct Mean (SD)	*F*	*η2*
Semantic similarity (LSA) response and input (rare word dominated)	0.362 (0.159)	0.458 (0.111)	0.499 (0.079)	102.799**	0.131
Semantic similarity (LSA) response and input (frequent word dominated)	0.403 (0.155)	0.5 (0.113)	0.531 (0.096)	95.432**	0.122
Academic frequency COCA function words	24524.248 (16585.406)	36788.308 (13168.904)	34324.442 (11401.743)	76.716**	0.101
Written frequency (BNC) function words	1.000 (0.441)	1.227 (0.291)	1.25 (0.256)	53.237**	0.072
Percentage of adjectives	0.086 (0.082)	0.112 (0.069)	0.135 (0.074)	38.42**	0.053
Academic range (COCA) all words	-0.494 (0.254)	-0.401 (0.114)	-0.411 (0.096)	24.093**	0.034
Number of words	24.417 (29.134)	33.476 (53.923)	38.618 (39.975)	16.736**	0.024
Range (SUBTLEXus) content words (no key words)	3737.317 (1693.106)	3227.84 (1437.09)	3213.191 (1105.223)	15.819**	0.023
Academic frequency (COCA) content words sentence minimum	0.743 (0.705)	0.941 (0.532)	0.922 (0.487)	12.386**	0.018
Word familiarity (MRC) sentence minimum	497.207 (206.379)	560.031 (126.208)	529.915 (165.372)	10.534**	0.015
Percent content words	0.635 (0.147)	0.597 (0.085)	0.606 (0.091)	9.621**	0.014
Word familiarity (MRC) content words (no key words)	465.777 (132.451)	483.335 (87.545)	495.526 (77.668)	6.393*	0.009
Range (COCA all words sentence minimum)	-1.937 (0.143)	-1.96 (0.083)	-1.956 (0.08)	4.063*	0.006
Academic range (COCA; no key words)	0.712 (0.081)	0.693 (0.076)	0.689 (0.137)	3.865*	0.006

* $p < .05$, ** $p < .001$

Table 2. Confusion matrix for DFA results for classifying scored responses

		Incomplete/incorrect	Partially correct	Correct	F_1 score
Whole set	Incomplete/incorrect	**605**	202	138	0.755
	Partially correct	31	**119**	60	0.400
	Correct	21	67	**129**	0.474

		Incomplete/incorrect	Partially correct	Correct	F_1 score
LOOCV	Incomplete/incorrect	**603**	203	139	0.752
	Partially correct	33	**113**	64	0.379
	Correct	22	70	**125**	0.459

reported by CRAT, 96 of these indices were normally distributed and not multi-collinear with one another. Of these 96 indices, 85 of the indices reported significant differences in the MANOVA analysis. These indices were related to overlap between the constructed response and the input received in the tutor, lexical sophistication, response length, response descriptiveness, and percentage of content words in the response. These indices were used in the subsequent DFA.

3.2 Discriminant Function Analysis

A stepwise DFA using the 85 indices selected through the MANOVA retained 14 variables related to semantic overlap between response and input, text descriptiveness, lexical sophistication, response length, and the use of content words. The indices retained in the DFA along with their means, standard deviations, *F* scores, *p* values, and effect sizes are reported in Table 1.

The results demonstrate that the DFA using these 14 indices correctly allocated 853 of the 1372 student responses in the total set, $\chi 2$ (df=4) = 393.169 $p < .001$, for an accuracy of 62.2%. For the leave-one-out cross-validation (LOOCV), the discriminant analysis allocated 841 of the 1372 texts for an accuracy of 61.3% (see the confusion matrix reported in Table 2 for results and F_1 scores). The Cohen's Kappa measure of agreement between the predicted and actual class label was 0.404, demonstrating moderate agreement.

4. DISCUSSION

This analysis provides an initial assessment of the extent to which the linguistic indices reported by the Constructed Response Analysis Tool (CRAT) are predictive of constructed responses. We examined student constructed responses to a single question in the ChemVLab+ system related to stoichiometiry. We found that 86 CRAT indices demonstrated differences between the three levels of human ratings (incomplete/incorrect, partially correct, and correct) and 14 of these variables were significant predictors of human scores in a DFA with a reported accuracy of 62%. The results suggest that the CRAT tool can be used to automatically classify student constructed responses based on human ratings of response accuracy. While preliminary, the results support the use of NLP tools in constructed response scoring and point toward specific linguistic features that can be used to predict human ratings of accuracy for student constructed responses.

The discriminant function analysis indicated that the strongest predictors of human accuracy scores were related to semantic similarity between the constructed response and the knowledge space provided (i.e., the available student hints in the ChemVLab+). The results indicated that student responses that had a higher semantic overlap with the hints were more likely to be correct or partially correct. These results held for rare word and frequent word LSA overlap. This suggests that students whose responses better represent the semantic space of the domain are more likely to produce correct responses.

Beyond semantic overlap with the hints, the next strongest predictors of human scores of student responses were related to the frequency of function words. These indices indicated that students who used more frequent function words were rated as having higher response scores (for both academic and written frequency). This likely indicates that students who used function words that occur more frequently in written contexts (i.e., academic writing and writing in general) construct more accurate responses. Thus, more successful students were those who were more likely to use writing styles frequent in academic English.

More successful answers also differed in the properties of the words they contained. More accurate answers were more descriptive in that they contained a greater number of adjectives. Though longer, successful answers contained fewer content words (i.e., they contained more function words). Successful answers contained more specific words (i.e., words that demonstrated a lower range score) and also contained more familiar and frequent words.

The model developed in this pilot study reports a level of accuracy that is appropriate to provide automated feedback to users in a tutoring system such as ChemVLab+. This feedback could include a summative score to provide users with an overall assessment of the quality of the constructed response. In addition, the model could be used to provide formative feedback to users in terms of language use (i.e., the use of academic language) and appropriate content (i.e., is writer covering the content of the question appropriately). Such feedback could be used by students to revise their responses and engage more deeply with the system. However, we would caution against using the reported model in high stakes assessments where accuracy is at a premium, although this advice should be empirically tested on a number of high stakes test corpora.

CRAT differs from many other scoring systems in that it is domain specific. Domain specificity has advantages as many of the key word and semantic indices can be trained on targeted content that increases construct validity and ensures that topic adherence on the part of the student remains an important component of constructed response scoring. Training the system, however, requires source texts that provide background about the topic. In some cases, these texts may be difficult to transfer to text files (in the case of lectures) or they may not exist within a system, limiting the generalizability of CRAT across a number of system.

Lastly, it remains an open question if a model trained on one area of chemistry will transfer to another area of chemistry or to domains outside of chemistry. For instance, the model developed here needs to be tested on similar but not overlapping chemistry topics and questions to test the model's generalizability within a macro-domain (e.g., with chemistry questions that address molecular equilibrium and acid bases). In addition, the model should be tested on domains outside of chemistry to assess whether constructed responses in various domains can be accurately scored based on a combination of semantic and keyword overlap between the response and the source and the use of academic language by system users.

5. CONCLUSION

This study introduces a freely available tool for constructed response scoring and tests the tool on a dataset of chemistry responses collected in the ChemVLab+. The findings indicate that the Constructed Response Analysis Tool (CRAT) can differentiate and classify student responses based on semantic overlap with text input, syntactic categories, text length, and lexical sophistication indices. Overall, the findings suggest that successful student responses contain greater overlap with the content learned and use more academic function words, more words in general, more descriptive words, and more familiar and frequent words that are also more specific.

Additional studies will be conducted to refine and continue to develop CRAT. For example, a future direction includes assessing the value of including indices of semantic overlap that use Latent Dirichlet allocation (LDA) spaces, allowing for topic modeling along with semantic graph analyses. CRAT also needs to be tested on additional constructed responses, including responses from a variety of domains. Lastly, the models developed using the CRAT tool should be assessed for application in providing feedback to users in instructional systems. Such follow up studies will provide additional information about the reliability of CRAT and the linguistic features within CRAT that are predictive of human ratings of constructed responses within different domains and on-line learning environments.

6. ACKNOWLEDGMENTS

This research was supported in part by the Institute for Education Sciences and National Science Foundation (IES R305A080589, IES R305A100069, IES R305G20018-02, DRL -1418072, and DRL- 1418378). Ideas expressed in this material are those of the authors and do not necessarily reflect the views of the IES or the NSF.

7. REFERENCES

[1] NGSS Lead States. 2013. *Next Generation Science Standards: For States, By State*s. Washington, DC: The National Academies Press.

[2] Liu, O. L., Brew, C., Blackmore, J., Gerard, L., Madhok, J., & Linn, M. C. (2014). Automated Scoring of Constructed-Response Science Items: Prospects and Obstacles. *Educational Measurement: Issues and Practice*, *33*(2), 19-28.

[3] Wainer, H., & Thissen, D. (1993). Combining multiple-choice and constructed response test scores: Toward a Marxist theory of test construction. *Applied Measurement in Education, 6,* 103–118.

[4] Williamson, D., Xi, X., & Breyer, J. (2012). A framework for evaluation and use of automated scoring. *Educational Measurement: Issues and Practice*, 31(1), 2–13.

[5] Honey, M. A., & Hilton, M. (Eds.). (2010). *Learning science through computer games and simulations*. Washington, DC: National Academies Press.

[6] Mitrovic, A. (2012). Fifteen years of constraint-based tutors: what we have achieved and where we are

going. *User Modeling and User-Adapted Interaction*, *22*(1-2), 39-72.

[7] Attali, Y., & Powers, D. (2008). Effect of immediate feedback and revision on psychometric properties of open-ended GRE subject test items. GRE Board Research Rep. No. 04-05; ETS RR-08-21. Princeton, NJ: Educational Testing Service.

[8] Bennett, R. E., & Sebrechts, M. M. (1996). The accuracy of expert-system diagnoses of mathematical problem solutions. *Applied Measurement in Education*, 9, 133–150.

[9] Wang, H.-C., Chang, C.-Y., & Li, T.-Y. (2005). Automated scoring for creative problem-solving ability with ideation-explanation modeling. *In Proceedings of the Thirteenth International Conference on Computers in Education* (pp. 522–529). Singapore: IOS Press.

[10] Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, *25*(2-3), 259-284.

[11] Brysbaert, M., Warriner, A.B., & Kuperman, V. (2013). Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*. doi:10.3758/s13428-013-0403-5

[12] Coltheart, M. (1981). The MRC psycholinguistic database. *Quarterly Journal of Experimental Psychology, 33A*, 497-505. doi:10.1080/14640748108400805

[13] British National Corpus, version 3 (BNC XML ed.). (2007). Retrieved from http://www.natcorp.ox.ac.uk

[14] Davies, M. (2010). The Corpus of Contemporary American English as the first reliable monitor corpus of English. *Literary and linguistic computing*, *25*(4), 447-464.

[15] Kyle, K. & Crossley, S. A. (2015). Automatically assessing lexical sophistication: Indices, tools, findings, and application. *TESOL Quarterly 49*(4), pp. 757-786. doi: 10.1002/tesq.194

[16] McNamara, D. S., Cai, Z., & Louwerse, M. M. (2007). Optimizing LSA measures of cohesion. *Handbook of latent semantic analysis*, 379-400.

Choosing versus Receiving Feedback: The Impact of Feedback Valence on Learning in an Assessment Game

Maria Cutumisu
University of Alberta
6-102 Education North, Edmonton, AB T6G 2G5
(780) 492 5211; cutumisu@ualberta.ca

Daniel L. Schwartz
Stanford University
485 Lasuen Mall, Stanford, CA 94305
(650) 725 5480; danls@stanford.edu

ABSTRACT
Studies examining feedback in educational settings have largely focused on feedback that is received, rather than chosen, by students. This study investigates whether adult participants learn more from choosing rather than receiving feedback from virtual characters in a digital poster design task. We employed a yoked study design and two versions of an online game-based assessment, Posterlet, to compare the learning outcomes of N=264 Mechanical Turk adults in two conditions: when they *chose* the feedback valence versus when they *received* the same feedback valence and order. In Posterlet, players design posters and learn graphic design principles from feedback. We found that the more the participants chose critical feedback, the more time they spent designing posters, but there were no differences in learning, revision, and time spent designing posters between conditions. In each condition, critical feedback correlated with performance and revision, suggesting that feedback valence is important for performance, regardless of being a choice.

Keywords
feedback valence, choice, assessment, game, learning

1. INTRODUCTION
A central goal of education is to prepare independent learners [16]. Previously, we operationalized this goal by a) identifying promising behaviors for autonomous learning that would reveal how students learned and b) creating novel choice-based digital assessment games that measured these behaviors. For instance, we measured students' choices to seek critical feedback and to revise, and we found that students who were more willing to seek critical feedback also learned more [4]. We examine learning choices (e.g., seeking social feedback), because such learning strategies can support ongoing learning, adapting to new challenges, and, ultimately, learning *how* to learn. These types of design thinking competencies, together with collaboration, persistence, and creativity, are crucial for 21st-century challenges, yet they are not formally assessed in schools [1, 21]. There are two main reasons why we need to measure learning behaviors. First, learning behaviors or attitudes enable learners to solve problems even when they do not have the domain knowledge skills to do so (e.g., collaborate with a partner from a different discipline). Second, current self-assessment techniques are not gender neutral: even though women and men scored similarly on a science exam (they had similar skills), women underestimated while men overestimated their performance (their attitudes did not match their skills; [7]). Such self-regulated learning behaviors [10] are worth investigating because revised self-assessment interventions may increase female representation in science, technology, engineering, and mathematics and could help create gender-inclusive 21st-century learning and assessment environments.

We previously examined the feedback valence (i.e., critical versus confirmatory) and its impact on performance and learning. In this study we examine for the first time the effect of feedback agency (i.e., choosing versus receiving). Our objective is to investigate the effect of choosing versus receiving feedback on learning, by comparing learning outcomes between participants who choose feedback and those who receive the same amount, valence, and order of feedback. We outline related work and theoretical perspectives that guide our research. Then, we describe our assessment environment, Posterlet, an online game designed to collect and assess participants' feedback and revision choices. We also created and presented a modified version of this game to accommodate the situation in which feedback is assigned to the learner in a principled way that mirrors the feedback chosen in the original Posterlet version. We then present evidence of the impact of choosing *versus* receiving feedback on learning outcomes, as well as theoretical and practical implications of this research.

We examine the impact of feedback choice and valence on learning by posing the following research questions:

1) Does critical feedback correlate with learning outcomes?
2) Are there learning outcome differences between choosing and receiving feedback?
3) Are there design duration differences between choosing and receiving feedback?
4) Are there gender differences on the measures by condition?

2. RELATED WORK
We distinguish several themes in the literature related to the theoretical perspectives that guide this research.

Choice-based Assessments. Traditional assessments measure learners' knowledge at the end of instruction, focusing on knowledge accuracy but providing little information about learners' readiness to learn new things. Vygotsky highlighted the importance of measuring learning processes [23], rather than only learning outcomes, to achieve deeper insights into students' potential to learn on their own. Schwartz and Bransford advocated *preparation for future learning* (PFL) assessments [19], which create learning opportunities during the assessment. Our research draws from work on *constructivist assessments* [20] and *choice-based assessments* [18]. Both these assessments build upon PFL assessments and measure not only learners' knowledge outcomes but also their learning processes (e.g., choices about what, when, and how to learn). For example, Posterlet [4], an online game that collects players' choices to seek critical feedback and to revise while they design posters, constitutes an instance of a choice-based assessment. The design of Posterlet is guided by the three core principles of choice-based assessments: *typical performance* (assessments need to capture every-day learning behaviors, not

test performance), *PFL* (assessments need to offer learning opportunities with measurable outcomes; [2]), and *choice* (assessments need to collect free learning choices that do not hinder the learners' ability to complete the assessments). Specifically, Posterlet provides players with a 10-15 minute fun game experience, with a chance to learn graphic design principles and to safely explore choices to seek critical feedback and revise, before applying them in more high-stakes situations. Concomitantly, Posterlet provides researchers with a way to track players' behaviors and learning outcomes to infer how prepared players are to learn on their own in new learning situations.

Confirmatory versus Critical Feedback. In educational contexts, feedback is defined as information related to a person's performance or understanding [11] and it is predominantly assigned by a teacher or a computer rather than chosen by the learner. There are some exceptions, but they pertain to help seeking [17] rather than specifically to feedback seeking. Here, we are mainly interested to investigate whether being given a choice about how to learn (i.e., choosing versus receiving feedback) has any impact on learning outcomes and other learning behaviors. In addition to feedback choice, the feedback literature provides some indication of the importance of feedback valence. For instance, critical feedback yields mixed results for performance [13], but studies of organizations show that most new ideas need critical constructive feedback to become successful [15]. A first challenge is that feedback is often absent from ideation environments. A second challenge is that critical feedback is even more elusive in such environments and it runs the risk of ego threat that causes people to reject instead of heed the feedback [11]. This suggests that attitudes towards seeking critical feedback are worth exploring. However, there is no evidence that the choice of critical feedback is as important as simply assigning critical feedback to the learner. Thus, we designed a variation of Posterlet and we employed a reduced-length game version for comparison to address this issue.

Choosing versus Receiving Feedback. Traditionally, most studies focused on supervised feedback, where the teacher assigned feedback to the student. However, in many situations, people need to actively seek feedback. Little is known about the implications of students' feedback choices on their learning or about variables that influence students' feedback choices, but researchers acknowledge the importance of the mechanisms underlying feedback for learning. For instance, Zimmerman [24] included "responsiveness to self-oriented feedback" among three critical features of students' self-regulated learning strategies. The effect of actively choosing rather than passively receiving critical feedback for learning raises interesting psychological questions. For example, patients who had control over their level of pain medication chose lower doses than those prescribed by medical staff [12]. Similarly, having a choice over critical feedback may act as a buffer against ego threat. Further, if learners are assigned critical feedback, would that lead to less learning than if they chose it? Consumer research provides corroborating evidence directly relevant to our prior research regarding the choice between confirmatory and critical feedback. Researchers found that novices sought confirmatory feedback more often, whereas experts sought critical feedback more often [9]. However, in contrast to our research, they did not measure learning outcomes.

3. POSTERLET

We employed two versions of the Posterlet game [4] to carry out our experiment. Participants playing the games assumed the identity of a school committee member in charge with designing a poster for each of the two booths advertising events for the school's Fun Fair. The effectiveness of each designed poster (i.e., the number of visitors attracted by the booth) is quantified by the number of tickets sold, which is displayed when the poster is submitted. Posterlet also measures the number of times critical feedback is chosen or received, depending on condition, and the player's choices to revise posters across the game. After designing each poster, the player chooses three virtual characters out of a focus group to find out what they think about the poster. In the Choose condition, the player clicks on one box ("I like" or "I don't like") above each character. For example, in Figure 1, a participant in the Choose condition has first selected critical feedback from the lion and then confirmatory feedback from the elephant, but no feedback from the panda yet.

Figure 1. In the Choose condition, the player has first chosen critical feedback from the lion, confirmatory feedback from the elephant, and no feedback from the panda yet.

In the Receive condition, the player clicks on the "Click for feedback" box to reveal a feedback valence assigned by the game. For example, in Figure 2, a Receive condition participant has first clicked on the elephant's "Click for feedback" box (revealing critical feedback), then on the ostrich's "Click for feedback" box (revealing confirmatory feedback). The amount of critical feedback chosen or assigned (depending on the condition) is Posterlet's first key measure. After reading the feedback, the player has a choice to revise or submit the poster. The number of revised posters is Posterlet's second key measure. The game's feedback system generates feedback by analyzing each poster against 21 graphic design principles provided by a graphic artist and organized into three broad categories: information (e.g., the poster should include the date of the event), readability (e.g., the color contrast between the text and the background should be high), and space use (e.g., the space used by images needs to be within 30% and 70% of the poster's surface).

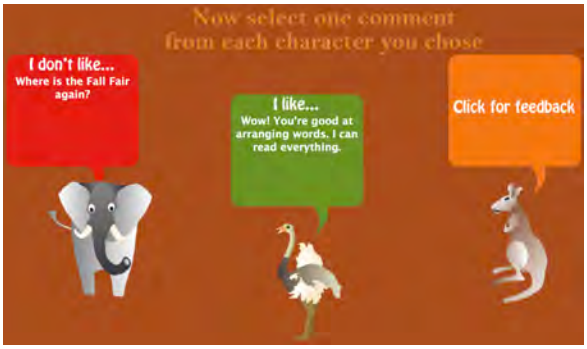


Figure 2. In the Receive condition, the player has first clicked on the elephant and received critical feedback, then on the ostrich and received confirmatory feedback.

It computes each poster's quality (i.e., the number of tickets sold) and it includes a priority scheme to ensure a balanced representation of these categories in the feedback. The critical and confirmatory feedback phrases are equivalent in length and informational content. For example, if a player omits the day of the fair, the critical feedback is: "You need to tell them what day the fair is." Otherwise, the confirmatory feedback is: "It's good you told them what day the fair is.", as shown in Figure 2.

4. METHOD

4.1 Participants, Procedures, Data Sources, and Experimental Overview

Participants (see Table 1) are N=264 Mechanical Turk adults randomly assigned to either the Choose or the Receive condition. Choose condition participants played a version of Posterlet that collected their feedback choices, while Receive condition participants played a modified Posterlet version that did not offer a feedback choice. In a one-to-one yoked experimental design, each participant in the Receive condition was assigned the feedback valence, number, and order of the feedback chosen by a matched Choose condition participant. Participants played a two-poster version of the Posterlet game individually, corresponding to their assigned condition, with a five-minute time limit on each poster or revision. Then, they completed an individual online posttest. The participants in the Choose condition were presented with a choice regarding the valence of their feedback. For instance, Figure 1 illustrates the feedback choices of a participant in the Choose condition: the participant chose a critical feedback from the lion and then a confirmatory feedback from the elephant. The Receive Condition participants were assigned the feedback valence of paired Choose condition participants, in the same order in which feedback was chosen by those paired participants. The game also collected participants' revision choices and computed the participants' poster performance (i.e., the quality of all their posters). Posterlet tracked the amount of critical feedback out of a maximum of 6 (3 feedback opportunities x 2 posters), as well as the amount of revisions out of a maximum of 2 (1 revision opportunity x 2 posters). A separate posttest measured the graphic design principles learned by participants in both conditions.

Table 1. Number of participants in each condition by gender

Cond.	Gender		Age Range	M_{age} (SD_{age})
	F	M		
Choose	54	78	19-69	32.26 (9.53)
Receive	61	71	19-63	33.30 (10.40)
Total	**115**	**149**	**19-69**	**32.78 (9.96)**

For instance, Figure 2 illustrates the feedback selection of a participant in the Receive condition: the participant was first assigned critical feedback and then confirmatory feedback, just like the participant in the Choose condition illustrated in Figure 1.

In the Choose condition, participants played Posterlet for an average of M=7 minutes (SD=3.11) and then completed the posttest for an average of M=6 minutes (SD=2.24). In the Receive condition, participants played Posterlet for an average of M=7 minutes (SD=2.91) and then completed the posttest for an average of M=7 minutes (SD=2.54). This study is correlational and experimental, aiming to determine whether having a choice about one's feedback valence aids in learning or in choosing to revise one's work. It compares adults who exercised a choice regarding the valence of their feedback (Choice condition) to adults who were assigned their feedback valence (Receive condition).

4.2 Dependent Measures

4.2.1 Feedback Valence and Revision Choices

Critical Feedback measures the number of "I don't like" boxes chosen or received by the player across the game (0-6). **Confirmatory Feedback** measures the number of "I like" boxes chosen or received, equivalent to 6 minus *Critical Feedback* (0-6), since there are six total feedback choices across the game. **Revision** measures the number of posters a player revised (0-2).

4.2.2 Design Duration

We measured the time a participant spent designing each poster, from the moment a booth theme was clicked to the moment the "Test" button was pressed.

4.2.3 Learning Outcomes

Poster Quality measures the poster performance, summing the poster quality across posters. The quality of each poster is the sum of the scores for each of the 21 features: 1 if a feature is always used correctly, 0 if a feature is not on the poster, and -1 if a feature is used incorrectly. Thus, the score of any individual poster ranges from -21 to 21, while *Poster Quality* from -42 to 42.

A posttest assessed learning of the graphic principles. The overall *Posttest* score represents the sum of the normalized scores of the *Recognition* and *Principle Selection* measures.

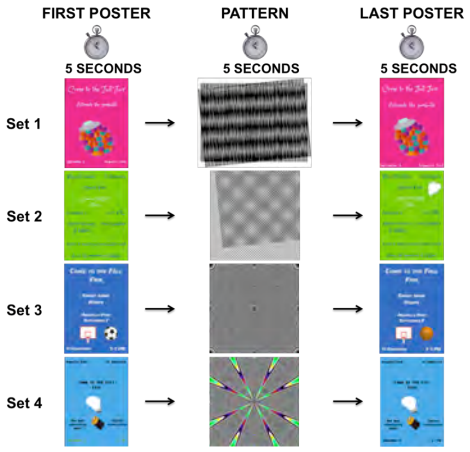


Figure 3. The *Recognition* posttest questions.

Recognition comprised four sets of posters (Figure 3). For each set, participants' task was to judge whether the quality of the second poster was the same/better/worse compared to the quality of the first poster and to provide a brief written explanation for their decision. A distractor image was inserted between the two posters to ensure that memory was not playing a role [22]. Participants were guided through a mini-tutorial and a trial poster comparison, in which pictures succeeded automatically on a five-second timer. Each correct answer is scored with one point, while each incorrect answer is scored with zero points. This measure sums up only the correct answers, thus ranging from zero to four. **Principle Selection** comprised two 10-item design principle checklist questions (Figure 4). A point was awarded/subtracted for each correct/incorrect answer and scores were summed up.

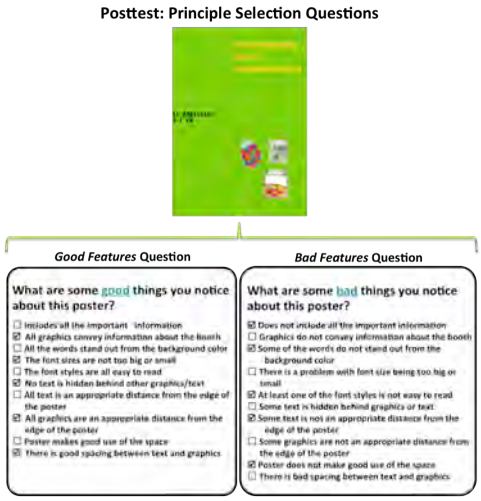


Figure 4. The Principle Selection posttest questions.

5. RESULTS

5.1 Does critical feedback correlate with learning outcomes?

We examined poster performance and design principle learning. Table 2 and Table 3 show the zero-order Pearson correlations by condition. Critical Feedback and Revision correlated with Poster Quality and strongly with each other. We consider Poster Quality a learning measure, due to participants' improvement across the game [*Choose*: round$_1$=10.64 (SD=5.0), round$_2$=11.76 (SD=4.5), Wilks' Lambda=.92, partial eta squared=.08, F(1,131)=11.67, p<.01; *Receive*: round$_1$=10.68 (SD=6.0), round$_2$=11.67 (SD=5.4), Wilks' Lambda=.96, partial eta squared=.04, F(1,131)=5.89, p<.05]. Revision correlated with Posttest and Design Duration. Poster Quality correlated with Posttest, supporting the learning measures' internal validity. In the Choose condition, Critical Feedback correlated with Design Duration.

Table 2: Correlations between critical feedback, revision, and learning outcomes for the *Choose* condition

Measures (N=132)	Revision	Poster Quality	Posttest	Design Duration
Critical Fb.	**.62****	**.25****	.08	**.32****
Revision	--	**.23****	**.21***	**.39****
PosterQuality		--	**.27****	**.39****

** p < .01, * p < .05

Table 3: Correlations between critical feedback, revision, and learning outcomes for the *Receive* condition

Measures (N=132)	Revision	Poster Quality	Posttest	Design Duration
Critical Fb.	**.58****	**.18***	.13	.16
Revision	--	**.24****	**.21***	**.36****
PosterQuality		--	**.21***	**.38****

** p < .01, * p < .05

We entered Critical Feedback and Revision in regressions to determine if they were independent predictors of the learning outcomes. In the Choose condition, for Poster Quality, the model was significant [F(2,129)=5.10, p<.01, R^2=.07, Adjusted R^2=.06], but Critical Feedback [t(129)=1.6, p=.11] and Revision [t(129)=1.6, p=.25] were not predictors. For Posttest, the model was significant [F(2,129)=3.33, p=.04, R^2=.05, Adjusted R^2=.03], Revision was a predictor: t(129)=2.38, p=.02, but Critical Feedback: t(129)=-.71, p=.48 was not. In the Receive condition, for Poster Quality, the model was significant [F(2,129)=4.23, p=.02, R^2=.06, Adjusted R^2=.05], Revision was a marginally significant predictor: t(129)=1.99, p<.05, but Critical Feedback: t(129)=.58, p=.56 was not. The Posttest model was not significant.

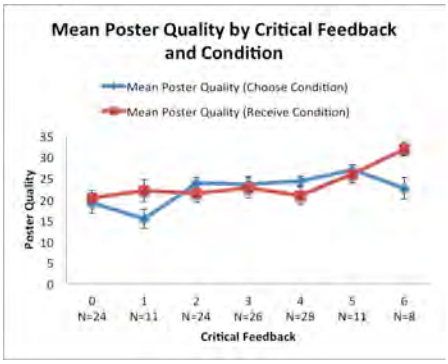


Figure 5. Poster Quality by Critical Feedback and condition.

5.2 Are there learning outcome differences between choosing and receiving feedback?

T-test analyses revealed no differences in Poster Quality [M_{Choose}=22.39 (SD=8.71), $M_{Receive}$=22.36 (SD=10.4), t(262)=.03, p=.97], Posttest [M_{Choose}=.10 (SD=1.53), $M_{Receive}$=.04 (SD=1.45), t(262)=.32, p=.75], and Revision [M_{Choose}=.80 (SD=.87), $M_{Receive}$=.93 (SD=.82), t(262)=-1.24, p=.22] between conditions. Figure 5, Figure 6, and Figure 7 plot our measures across the game as a function of critical feedback (from 0 to 6) by condition. Error bars represent one standard error. The x-axis shows the range of critical feedback and the number of participants for each amount of critical feedback (e.g., N=26 participants chose/received 3 pieces of critical feedback across all posters). Regressions of *critical feedback*, *condition*, and *critical feedback by condition* on learning and revision revealed no interactions of critical feedback and condition with our measures.

5.3 Are there design duration differences between choosing and receiving feedback?

A t-test analysis revealed no differences in Design Duration (time in seconds spent designing posters) between conditions [M_{Choose}=401.30 (SD=186.39) and $M_{Receive}$=394.44 (SD=174.94), t(262)=.31, p=.76]. Figure 8 plots participants' poster design time across the game as a function of critical feedback (from 0 to 6) by condition.

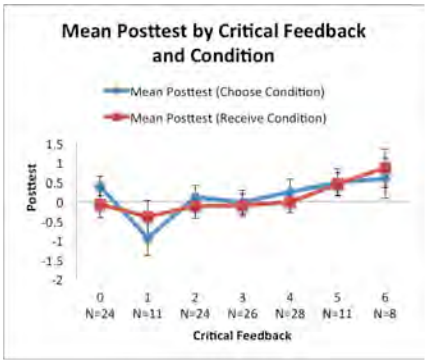


Figure 6. Posttest by Critical Feedback and condition.

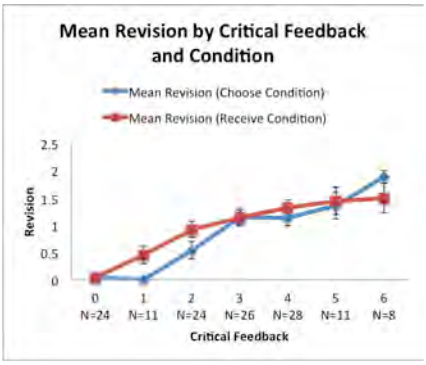


Figure 7. Revision by Critical Feedback and condition.

5.4 Are there any gender differences?

In the Receive condition, we found that females [M=433.28 (SD=176.84), $t(130)=2.41$, $p=.02$] spent more time designing posters than males [M=361.07 (SD=167.40)]. There were no gender differences by condition on any of the rest of the measures (Revision, Poster Quality, and Posttest).

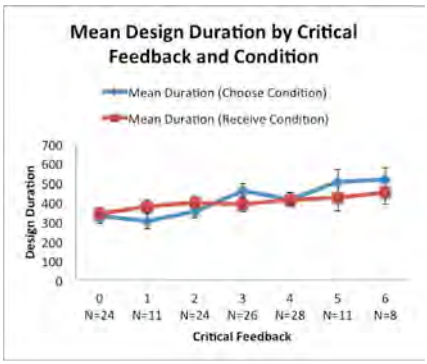


Figure 8. Design Duration by Critical Feedback and condition.

6. DISCUSSION

This is a first-of-kind examination of both the agency (choosing versus receiving) and the valence (critical versus confirmatory) of feedback and their impact on performance and learning. We found that, in each condition, the amount of critical feedback (either chosen or received) correlated with participants' performance on the poster design task. Consistent with our previous findings [3, 4], critical, rather than confirmatory, feedback seems beneficial for learning. Also, the choice to revise was beneficial for performance and learning outcomes and it strongly correlated with critical feedback (chosen or received). We found no differences between conditions in any of the measures outlined in this paper. These results held when we compared the measures by gender in each condition, although in the Receive condition, females spent more time designing posters than males. This indicates that these types of behavioral assessments of learning have the potential to be gender neutral. The next step would be to design more such dynamic assessments to evaluate other behaviors, such as self-assessment. Designing gender-neutral assessments that embed both skills and learning behaviors would bring us closer to determining the knowledge, skills, and delivery methods required to foster independent learners in the 21st century, as well as ways to ensure gender equality, especially when only 14.1% of North American computer science bachelor's degree graduates are female [25]. Our study points to critical, rather than confirmatory, feedback being beneficial for learning, regardless of being chosen or assigned. It also points to ways of designing assessments that measure learning behaviors equally regardless of gender. Finally, in the Choose condition, the more the participants chose critical feedback, the more time they spent designing posters. The relation between critical feedback and revision, as well as between critical feedback and poster quality, was stronger and more stable in the Choose condition, pointing to motivational factors of choosing versus receiving critical feedback for performance. More research is needed to elucidate this motivational aspect.

People's choices of critical feedback can be influenced by a wide range of factors. For instance, the perception of a trait as fixed may lead to avoidance of negative feedback [5]. Additionally, compared to a growth mindset (an incremental theory of intelligence - the belief that intelligence can be developed over time), a fixed mindset (an entity theory of intelligence – the belief that intelligence is fixed) was found to be associated with decreased attention to corrective feedback or errors [14]. However, the results of this study suggest that there is no underlying variable (e.g., desire to learn, self-confidence, growth mindset [6, 8], etc.) that drives the effect of critical feedback. People who choose critical feedback more often may exhibit one or more of these variables, yet, despite that, assigning the same amount of feedback leads to the same results as other factors that may causes them to choose critical feedback. Consequently, it seems that such factors (e.g., deep beliefs or personal attributions, such as "I am a learner") do not need to be changed to help people reap the benefits of constructive criticism. Learner beliefs do not mediate the benefits of receiving constructive criticism. One potential implication is the possibility to change people's beliefs about seeking critical feedback without having to change their broad beliefs about themselves as learners, which we also demonstrated in a separate study [3]: fairly straightforward instruction to seek social feedback (i.e., opinions of others) transferred to Posterlet and, consequently, students learned more.

Our study's limitations are associated with conducting Mechanical Turk experiments with a large population: (1) a maximum of five minutes allotted per poster, which may have hindered the discovery of some of the game's features (e.g., that the poster background color can be changed) and (2) a maximum of two game levels, which offered participants at most six pieces of feedback from which to learn graphic design principles, which may not have overlapped with the four principles included on the posttest (feedback content varied, depending on each participant's poster, but the posttest questions were the same for all participants). The latter is one possible explanation for the lack of correlation between critical feedback and posttest. Alternatively,

participants examined each poster for only five seconds and, if they missed one of the two posters in a set, they could not have accurately answered any of the questions about that set. Thus, we plan to compare this study's Choose condition data with data from the first two levels of previous three-level Posterlet game studies. That way, we may predict participant behaviors on the third game level, to potentially detect differences between conditions in our measures that are not apparent currently.

7. CONCLUSIONS

We modified a choice-based assessment game to measure learning when participants are offered a choice about the valence of their feedback and when they are assigned their feedback valence. The data enabled a novel examination of choosing versus receiving confirmatory *versus* critical feedback with regards to learning outcomes. We found that the more the participants chose critical feedback in the Choose condition, the more time they spent designing posters. There were no differences in learning outcomes (performance on the poster design task and learning of the graphic design principles), choice to revise, or time spent designing posters between participants who chose feedback and those who received the same amount, valence, and order of feedback. We plan a similar study with middle-school and college students to explore instruction and assessment implications. These studies could inform teachers to create environments in which students feel encouraged to engage more with critical feedback (proactively or reactively), even in open-ended tasks as digital poster design. The flexibility of such short assessments focused on specific choices (e.g., feedback seeking) enables the development and evaluation of a variety of instruction models. Concomitantly, researchers can design pedagogical interventions and learning environments that embed such assessments to empower all learners, regardless of gender, to be innovative, confident, and prepared for the challenges of the 21st century.

8. ACKNOWLEDGMENTS

We thank the Gordon and Betty Moore foundation and the NSF (Grant # 1228831), Jacob Haigh for assistance with the online setup, as well as the Mechanical Turk participants.

9. REFERENCES

[1] Black, P., & Wiliam, D. (2009). Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability, 21*(1), 5–31.

[2] Bransford, J. D., & Schwartz, D. L. (1999). Rethinking transfer: A simple proposal with multiple implications, *Review of Research in Education, 24*, 61–100.

[3] Conlin, L., Chin, D. B., Blair, K. P., Cutumisu, M., & Schwartz, D. L. (2015). Guardian Angels of Our Better Nature: Finding Evidence of the Benefits of Design Thinking. In Proc. of *ASEE*, June 14-17, Seattle, WA, USA.

[4] Cutumisu, M., Blair, K. P., Chin, D. B., & Schwartz, D. L. (2015). Posterlet: A Game-Based Assessment of Children's Choices to Seek Feedback and to Revise. *Journal of Learning Analytics, 2*(1), 49-71.

[5] Dunning, D. (1995). Trait importance and modifiability as factors influencing self-assessment and self-enhancement motives. *Personality and Social Psychology Bulletin, 21*.

[6] Dweck, C. S., & Leggett, E. L. (1988). A social-cognitive approach to motivation and personality. *Psychological Review, 95*(2), 256-273.

[7] Ehrlinger, J., & Dunning, D. (2003). How chronic self-views influence (and potentially mislead) estimates of performance. *J of Personality & Social Psychology*, *84*(1), 5.

[8] Ehrlinger, J., Mitchum, A. L., & Dweck, C. S. (2016). Understanding overconfidence: Theories of intelligence, preferential attention, and distorted self-assessment. *Journal of Experimental Social Psychology*, *63*, 94-100.

[9] Finkelstein, S. R., & Fishbach, A. (2012). Tell me what I did wrong: experts seek and respond to negative feedback. *Journal of Consumer Research, 39*(1), 22–38.

[10] Garrison, D. R. (1997). Self-directed learning: Toward a comprehensive model. *Adult Ed. Quarterly*, *48*(1), 18–33.

[11] Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research, 77*(1), 81–112.

[12] Haydon, M. L., Larson, D., Reed, E., Shrivastava, V. K., Preslicka, C. W., & Nageotte, M. P. (2011). Obstetric outcomes and maternal satisfaction in nulliparous women using patient-controlled epidural analgesia. *American Journal of Obstetrics and Gynecology*, *205*(3), 271-e1.

[13] Kluger, A. N., & DeNisi, A. (1998). Feedback interventions: toward the understanding of a double-edged sword. *Current Directions in Psychological Science, 7*(3), 67–72.

[14] Mangels, J., Butterfield, B., Lamb, J., Good, C., & Dweck, C. (2006). Why do beliefs about intelligence influence learning success? A social cognitive neuroscience model. *Social Cognitive and Affective neuroscience*, *1*(2).

[15] March, J. (2008). *Explorations in organizations*. Stanford University Press.

[16] Piaget, J. (1964). Quoted by Eleanor Duckworth in "Piaget Rediscovered: A Report of the Conference on Cognitive Studies and Curriculum Development", *Cognitive Studies and Curriculum Development, New York*.

[17] Roll, I., Aleven, V., McLaren, B. M., & Koedinger, K. R. (2011). Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction*, *21*(2), 267-280.

[18] Schwartz, D. L., & Arena, D. (2009). Choice-based assessments for the digital age. *MacArthur 21st Century Learning and Assessment Project*.

[19] Schwartz, D. L., & Bransford, J. D. (1998). A time for telling. *Cognition and Instruction*, *16*, 475–522.

[20] Schwartz, D. L., Lindgren, R., & Lewis, S. (2009). Constructivism in an age of non-constructivist assessments. *Constructivist Instruction*, 34-61.

[21] Shute, V. J., & Ke, F. (2012). Games, learning, and assessment. In *Assessment in Game-Based Learning*, 43-58.

[22] Standing, L. (1973). Learning 10,000 pictures. *Quarterly Journal of Experimental Psychology, 25*, 207-222.

[23] Vygotsky, L. S. (1934). *The collected works of LS Vygotsky: Problems of the theory and history of psychology*.

[24] Zimmerman, B. J. (1990). Self-regulated learning and academic achievement: An overview. *Educational Psychologist, 25*(1), 3–17.

[25] Zweben, S., & Bizot, B. (2015). 2014 Taulbee Survey. *Computing Research News, May 2015, 27*(5), p. 20.

Course Content Analysis: An Initiative Step toward Learning Object Recommendation Systems for MOOC Learners

Yiling Dai
Graduate School of Informatics
Kyoto University
Yoshida-Honmachi, Sakyo-ku
Kyoto, Japan
daiyiling@db.soc.i.kyoto-u.ac.jp

Yasuhito Asano
Graduate School of Informatics
Kyoto University
Yoshida-Honmachi, Sakyo-ku
Kyoto, Japan
asano@i.kyoto-u.ac.jp

Masatoshi Yoshikawa
Graduate School of Informatics
Kyoto University
Yoshida-Honmachi, Sakyo-ku
Kyoto, Japan
yoshikawa@i.kyoto-u.ac.jp

ABSTRACT

With the accelerating development of open education, low-cost online learning resources, such as Massive Open Online Courses (MOOCs), are reaching a wide audience around the world. However, when faced with these appealing but overwhelming learning resources, learners are prone making rash learning decisions, which may be either excessive or insufficient to their learning capacities. To avoid the mismatch between learners and learning objects, we propose a supporting system that recommends a personalized path of learning objects for a given learner. In realizing this system, a domain knowledge structure is necessary to connect learners' information and learning objects. As an initiative step, we employ the Labeled Latent Dirichlet Allocation method to predict how the content of a course is distributed over different categories in the domain. We conduct experiments by utilizing course syllabi as course content, and curriculum guidelines as domain knowledge. The predicting performance is improved when involving external texts related to the concerned domain knowledge unit.

1. INTRODUCTION

Nowadays, pedagogically condensed free online resources are playing an increasingly more important role of facilitating self-learning. Among those resources, Massive Open Online Courses (MOOCs) are engendering a revolutionary change in higher education by distributing digital versions of university courses to everyone at a relatively low cost. Courses about Computer Science on Edx (one of the largest MOOC platforms), reached over 600,000 listeners during the period from 2012 autumn to 2014 summer [6], which hardly ever occurs on real campuses. However, compared with their popularity among audience, the low completion rate of courses (e.g, 7% of the MOOCs on Edx mentioned above) begs the question—how many learners have truly benefited from receiving MOOCs? It appears that MOOCs have a way to go to achieve its original goal of making education accessible to everyone.

Rather than not being able to receive traditional education, many users utilize MOOCs out of pure curiosity toward subjects, or to complement their academic lives or career development [2]. In addition, the occupations of MOOC users are diverse, from students, writers, and engineers to housewives [2]. This type of utilization of MOOCs sets a higher requirement in terms of learner's self-motivation and self-regulation. Consequently, many users have reflected that they did not have sufficient spare time to catch on to the process of MOOCs, or simply became stuck on the overwhelming learning contents [2].

An intuitive question concerning that how we can help to maintain this precious enthusiasm of refreshing one's knowledge, motives this paper. We hold the view that finding the "just right" learning objects for respective individuals paves the way toward a successful learning experience. This belief is also in agreement with the opinion of [4], which underlines the importance of personalization, especially in the context of online learning. Specifically, "just right" means that the learning objects fit both the learning objective and learning ability of a given learner. In the context of self-learning, where more flexibility is given to a learner for him to decide what to learn, the adaptation to learning objectives deserve greater investigation than before. Concerning the method used to accomplish personalization in learning, previous studies have shown a trend of utilizing expert manpower or learner performance data to extract internal relationships among knowledge itself and external relationships between knowledge and learner mastery, which may not work when promoting personalized learning on a massive scale.

In this paper, we propose the idea of a novel supporting system that automatically recommends an appropriate set of learning objects with cues of learning priority to a given learner. This system is expected to outperform existing

adaptive learning systems on addressing heterogeneous course materials automatically and on adapting learning objects to learners before they start to learn. As an initiative task, a course content analysis is conducted to crystallize the realization of the supporting system. We employ the Labeled Latent Dirichlet Allocation method to predict how the content of a course is distributed over different domain knowledge categories. Course syllabus texts are utilized as course content, and the knowledge listed in curriculum guidelines are utilized as domain knowledge. To improve the accuracy of predictions, we extend the content of the curriculum guideline by integrating external texts retrieved from search engines.

The remainder of this paper is structured as follows: Section 2 summarizes related work with regard to personalized learning and knowledge representation. In section 3, an illustration and the framework of the supporting system are sketched. Then, we present the results and observations of a course content analysis. Finally, we discuss on future work.

2. RELATED WORK

2.1 Personalized learning

What we call personalized learning is named differently in previous studies, e.g., adaptive learning/education, individualized learning/education, and intelligent tutoring systems; however, they all share the main concern of adapting learning materials to individual learners. In this paper, we adopt the phrase "personalized learning" to capture all these related studies and use "personalize", "individualize", "adapt" interchangeably.

Personalized learning is described as "learning tailored to the specific requirements and preferences of the individual" in [11]. Although not forming a fixed definition of personalized learning, many studies attempt to adapt learning to specific learners. [4] demonstrated a hypermedia textbook that can provide direct guidance and adaptive navigation support to learners. Similarly, [15] developed a topic-based adaptive learning system that directs the learner to the appropriate learning object by providing navigational cues. Moreover, [16] broadened the adaptation from a single source of personalization information to learning achievements and learning styles at one time. [8] presented an e-learning system that recommends learning items by detecting frequent learning sequences and similar learners. [9] proposed another approach of generating adaptive course content using concept filters.

A shared architecture of a personalized learning system that can be observed consists of three parts: Domain model, Learner model and Adaptation model. The domain model constructs all the knowledge units of learning materials in a common space, and its complexity varies based on the application contexts. The learner model is a projection of a learner's learning state (i.e., mastery level of knowledge, learning objective, and learning style) onto the structure of knowledge that is defined in the domain model. The adaptation model functions as a recommend of the next learning target basing on the updated learner state. This adaptation in learning environments occurs at different levels. [11] categorized this adaptation as follows: Adaptive Interaction, which occurs during the interactions between learners and the system; Adaptive Course Delivery, which intends to tailor learning materials to a given learner; Content Discovery and Assembly, which involve the collecting of learning materials from potential sources or repositories; Adaptive Collaboration Support, which supports communication in the learning process.

In the context of self-learning, "why I want to learn", "what I want to learn", "what outcomes I am expecting", things usually being told to the learner by the curriculum, must be determined by the learner himself. As a result, we consider that the information-seeking phase before starting to learn becomes a key to a successful learning experience. We provide a learning object recommendation system that the learners can resort to when they are faced with overwhelming learning resources. Compared with a branch of studies [10, 1, 19] that implement the adaptation by redirecting the learner to an optimal learning path using tracked learner performance, our approach focuses on a more macro level of adaptation, which occurs beforehand and addresses the learning object with a larger granularity (i.e., a lecture). According to [11]'s categorization of adaptation, our system stands in an overlapping area of Adaptive Course Delivery and Content Discovery and Assembly, thereby distinguishing itself from other adaptive learning/tutoring systems.

2.2 Automatic domain representation

The construction of domain knowledge is a key step in accommodating a personalized learning system. However, previous studies [4, 15, 16, 8, 9, 10, 1, 19] show a substantial reliance on expert efforts, whose systems require the instructors to define strictly structured course materials for the concerned system. This is so time-consuming and platform dependent that it is unsuitable when addressing a large amount of distributed learning materials. An automatic and interoperable knowledge representation and assemble are thus desired.

In the context of learning, knowledge representation refers to the process of editing knowledge in a more visually sound and retrievable manner based on its hierarchical or dependent relationships. Previous studies relating to this concept can be divided into two types according to their approaches, and we name them prior approaches and post approaches. A prior approach means extracting the relationships between knowledge units based on the structure defined by the instructor. For example, [3] utilized the content and structure of a textbook to extract the relationships between concepts based on their co-occurrence conditions. [5] exploited the extraction of prerequisite relationships of learning objects by conducting semantic analysis on Wikipedia articles. Regarding the post approach, in which the structure of knowledge is modified by the learner reactions on these learning objects, [17] and [18] attempted to detect prerequisite relationships between knowledge units by utilizing a considerable amount of learner achievement data. Their studies are based on the rationale that knowledge units that are statistically "always" mistaken by the learners should be learned before the ones that are not so.

In this paper, we emphasize the preprocess of learning (i.e., seeking information and making a learning plan), which occurs before a substantial amount of learner performance data

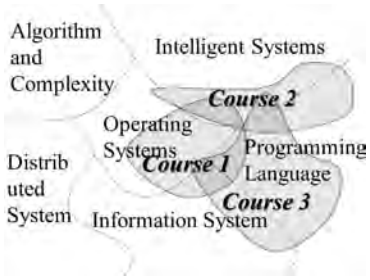


Figure 1: Illustration of our supporting system—the course map

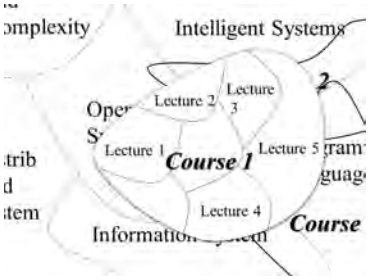


Figure 2: Illustration of our supporting system—the detailed course information

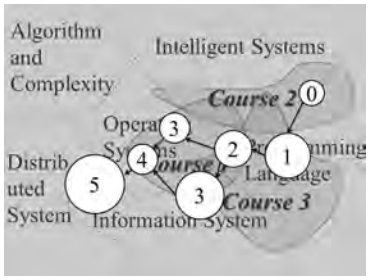


Figure 3: Illustration of our supporting system—a learning path

are available. Thus, our research falls into the category of prior approaches. Previous studies [3, 5] have employed various Natural Language Processing techniques to extract relationships between knowledge units. However, the results remain modest in addressing heterogeneous learning materials at scale; a proliferation of this stream of research is needed.

3. OUR SUPPORTING SYSTEM

As discussed in the previous section, in the context of self-learning, support for a learner determining what to learn and how to learn is sensible. Except for a learner's learning ability, which has received a fair discussion in previous research, we consider the estimation of the learner's learning objective. Regarding the level of personalization in this learning environment, we highlight the phase of assembling learning materials from distributed learning resources. As a consequence, we suppose that learners will benefit from our system before they enter the real learning process when offered a tailored path of learning objects that fits their learning needs and ability.

3.1 An illustration of the system

To explain our supporting system more vividly, we present an illustration of a final usage of the system. The target user of our system will not be constrained to a specific group of learners; however, the learners who will benefit the most from our system are those who are planning to challenge some unfamiliar subject. Then, we can imagine a virtual learner, a college student majoring in social science, who is wondering how data mining techniques will assist in analyzing his collected data.

First, he may simply input a keyword "data mining". Instead of returning a ranked list of relevant courses, which is normal in existing MOOC search engines, our system will answer the query dynamically by starting with a map of relevant courses to that query. As shown in Figure 1, the shapes circled using a dotted line with titles (e.g., "Intelligent Systems") on them refer to the predefined structure of the domain knowledge. In addition, the shape circled using a solid line represent a course that contains the knowledge in that place.

Then, the learner responds to the first reply differently. He may want to obtain details of some highly similar courses or seek a more holistic view of this domain to determine what these courses mean to his learning task. If the learner chooses to zoom in to course 1, then he will obtain a detailed view of the content of course 1. As shown in Figure 2, the topics covered in course 1 will be shown in the unit of a lecture.

We suppose that the learner will not be satisfied until he can make a confident decision on what and how to learn. Therefore, he will continue interacting with our system, during which time his learning characteristics will be recorded. Finally, the recorded learner information will be used to recommend a tailored learning path for the learner (see Figure 3). The path consists of a set of learning objects that are chained according to the dependent relationships between the knowledge they cover. For well-prepared learners, the path will exclude materials he already knows and will cover a narrowed down knowledge set in the depth. For novice learners, in this case, the path will cover a wider range of knowledge and will start from the very simple knowledge units.

3.2 The architecture of the system

To realize the system illustrated above, the architecture is threefold—domain model, learner model, and personalization model. The domain model conducts the task of locating the learning objects of courses in the knowledge structure of the domain. The learner model tracks learner information about his learning objective, background knowledge, and learning preferences according to the knowledge structure. The personalization model specifies the appropriate learning objects based on predefined criteria. Among them, the construction of domain knowledge and the mapping of course content determine how to estimate learner information and what learning objects to recommend. Thus, it is reasonable

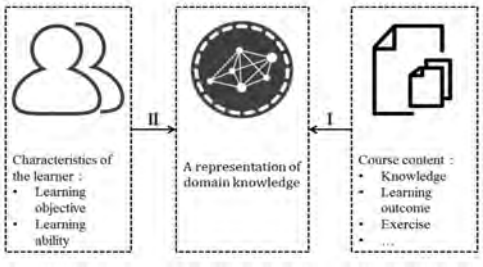


Figure 4: The architecture of proposed system

to exploit the domain model as a primary task. The following part of this paper describes a course content analysis and discusses its potential for equipping the domain model.

4. COURSE CONTENT ANALYSIS

4.1 Overview

As a primary task for matching course contents to a domain knowledge base, we extracted knowledge coverage of a given course by projecting its syllabus text onto a curricular guideline in the domain. A syllabus functions as a summary of the course content, which makes it suitable for our method. In addition, a curricular guideline generally contains important topics in the domain, which can be utilized as a reference of the domain knowledge. Specifically, we utilized the curriculum guideline *Computer Science Curricula 2013* (CS2013) [14] published by IEEE-CS and ACM, which attempts to provide instructional cues of knowledge that should be included in an undergraduate program. In CS2013, both classic and frontier topics in this domain are described in *Body of Knowledge* (BoK). BoK is compiled in a hierarchical structure wherein the smallest granularity of knowledge is a *topic*, and each *topic* belongs to a *Knowledge Unit* (KU), and each KU further belongs to a *Knowledge Area* (KA). In total, 18 KAs and 163 KUs are formed to categorize knowledge in the domain of Computer Science. A simplified example of KA-KU-Topic knowledge structure in CS2013 is shown in Table 1.

This semi-structured BoK has been used to analyze the curricula of different educational institutions [7, 13]. In an attempt to obtain an overall picture of Informatics programs in Japan, [7] conducted a judgement of knowledge coverage on syllabi by referring to curriculum guidelines. [13] employed a supervised Latent Dirichlet Allocation (LDA) method to extract KA coverages of a course using the text of its syllabus. From the above studies, it is reasonable to use curriculum guidelines as a knowledge base to form predictions of course knowledge coverage in an automated manner. However, it is not sufficient to recommend learning objects when solely using the knowledge coverage of a course at the level of KA. Therefore, we attempt to extract knowledge coverage of a course at a further fragmented level—KU in this case.

We adopt the topic model, Labeled Latent Dirichlet Allocation (Labeled LDA) to extract the knowledge coverage. Labeled LDA is designed to specify multiple dimensions of a given text that correspond to manually labeled tags [12]. In CS2013, exemplar courses with knowledge distribution information show that a course generally contains knowledge from more than one KA or KU. Therefor, this method is suited when addressing a syllabus text that is labeled with multiple predefined tags—KA/KU in this case.

Table 1: KA-KU-Topic knowledge structure in CS2013 [14]

KA	KU	Topics
Algorithms and Complexity (AL)	Basic Analysis	•Big O notation •...
	Algorithm Strategies	•Greedy algorithms •...
	...	•...

Table 2: An example of syllabus information in CS2013 [14]

What is covered in the course?
• The modeling process
• Two system dynamic tool tutorials
• Computational error
• ...

Considering that topics listed in BoK are highly compact representations of knowledge, we resort to external texts to complement the content of BoK. Specifically, we integrated snippet information retrieved from queries of a KU to improve the accuracy of predictions .

4.2 Dataset

81 exemplar courses, whose course information and knowledge distributions are assigned by the course instructor, are included. As shown in Table 2, the answer to the question "What is covered in the course?" is viewed as the syllabus information of a course. In addition, the information offered by the instructor on how the lecture hours of a course are allocated to each KA and KU is referred to as the ground truth of our method (e.g., 35.5 hours in CN, 3 hours in IS,...). After excluding malformed course information, 73 exemplar courses were used in the course content analysis.

Regarding the external texts, we threw 3 types of queries to retrieve snippet texts of websites from Google Custom Search API. The queries are formed by using: (1)KU title alone, (2) KA and KU title, (3) KU title and its top 3 representative terms (chosen by their tf-idf values, which represent an effective as an indicator of the importance of a term over a set of documents). 10 snippet texts were complemented to the content of each KU.

4.3 Procedures

4.3.1 Training set

As a trial analysis, we exploit the predictability of curriculum guidelines by conducting experiments with different training sets. Among all the experiments, 30 exemplar course syllabi were chosen randomly as the testing set. Concerning the training set, we set 2 variables, forming 8 patterns, to improve the accuracy of predictions. The first variable denotes whether manually labeled syllabus texts are used in the training set or BoK texts alone are used. The

Table 3: Experiment id

	BoK	BoK_Snippet1	BoK_Snippet2	BoK_Snippet3
BoK	KA-1-0	KA-1-1	KA-1-2	KA-1-3
BoK+Course Syllabus	KA-2-0	KA-2-1	KA-2-2	KA-2-3

second denotes what type of snippet texts are used, with "0" denoting using BoK texts alone.

Table 3 presents the naming of the experiments according to their content of the training set. The names of experiments for the prediction of KU knowledge coverage follow the same naming scheme. We conduct all 8 experiments on predicting knowledge coverage at the level of both KA and KU, and we add "KA" or "KU" to the experiment id to indicate the different targets.

4.3.2 Evaluation

To evaluate the predicted probabilities over KAs/KUs of a syllabus, we apply the Normalized Discounted Cumulative Gain (nDCG), which is used to evaluate the relevance of a document rank to a given query in classic Information Retrieval (IR). We choose the nDCG because it addresses relevance as a non-binary value, which is better suited to our case where the relevance of a document corresponds to lecture hours. For each course, we compare the ranked list of KAs/KUs that is predicted by our method, with the ranked list of KAs/KUs that is allocated by the course instructor. The computation is conducted using the following equations :

$$\begin{cases} G_c[i] = rel_c[i] \\ DCG_c[k] = \sum_{i=1}^{k} \frac{G_c[i]}{\log_2 i+1} \\ nDCG_c[k] = \frac{DCG[k]}{IDCG[k]} \end{cases} \quad (1)$$

Here, $rel_c[i]$ denotes the lecture hours allocated to the i^{th} KA/KU for a given course; DCG denotes the discounted cumulative gain of the ranked KA/KU list that is predicted by our method, and IDCG denotes the one of the ranked KA/KU list assigned by the course instructor.

4.4 Results

We utilized the *Stanford Topic Modeling Toolbox* to compute the KA/KU distributions of a syllabus and the Python library *Scikit Learn* to compute the tf-idf value of each term appearing in a BoK. Other data processes, such as the computation of the nDCG, are implemented in Python. Concerning the most representative terms for each KU, we chose the top three terms from a vocabulary of 2486 non-stopword terms. Because the average number of KAs that a course covers assigned by the instructor is 2.67, being 9.04 for KU, we focus on the nDCG value of $k = 3$ for KA, of $k = 9$ for KU. The results for each experiment are shown in Figure 5.

4.5 Discussion

As observed in Figure 5, all the nDCG values of the experiments with a training set containing BoK texts alone are higher than those with a training set consisting of both BoK texts and exemplar course syllabus texts. In our data set, all the BoK texts are annotated with one label, whereas exemplar course syllabus texts are annotated with multiple

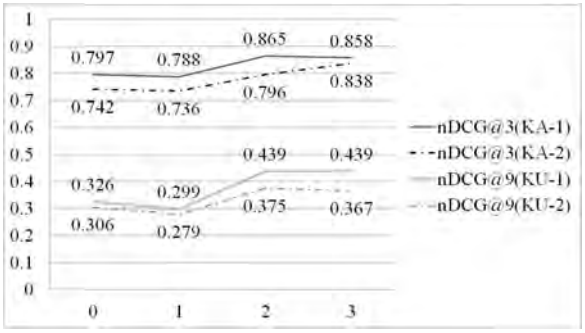


Figure 5: The nDCG values of each experiment. The vertical axis denotes the value of nDCG, which varies from 0 to 1. The horizontal axis denotes the second variable with regard to the naming of the experiments—the type of snippet texts used in training set.

labels. This unbalanced number of labels in the training set may reduce the precision of prediction obtained using Labeled LDA. However, from a positive perspective, this result indicates the potential of only using pre-collected documents of domain knowledge instead of collecting annotated course syllabi when predicting the knowledge coverage of a given course.

Two types of snippet texts exhibit a positive effect on predicting KA/KU knowledge coverage. They are snippet texts queried from KU titles with their corresponding KA title and snippet texts queried from KU titles with their top 3 representative terms. For example, nDCG@3 of KA-1-2 and KA-1-3 are notably higher than those of KA-1-0. A similar trend can also be observed in the case of predicting KUs. In contrast, nDCG@3 of KA-1-1 are lower than those of KA-1-0, which indicates that the external texts obtained from the KU title query drag down the performance of our model. One possible reason that can be inferred is that a sole KU title can produce substantial noise when it is used without context. For example, "processing" has a much broader meaning than that in the context of "Computational Science". Other ambiguous KU titles, such as "Basic Logic" and "Data, Information, and Knowledge", are prone to increasing the prevalence of this type of mistake. Overall, queries consisting of KA titles and KU titles or KU titles and their keywords provide effective and relevant texts when predicting knowledge coverage.

To seek deeper factors that may contribute to the correctness of a prediction, we examined an exemplar course syllabus and compared it with BoK and external texts. We found:

- Some synonymous or semantically similar phrases (e.g.,

"strategies for choosing..." and "apply...") may not be detected by our method.
- There exist internal relationships between KUs (e.g., KU "Processing" under KA "Computational Science" overlaps with KU "Algorithms and Design" under KA "Software Development Fundamentals"), which may mislead the prediction of KUs.
- An increase in performance in predicting KAs may not guarantee an improvement in predicting KUs. Because in some cases, the improvement in predicting KAs is achieved by assigning a probability to an incorrect KU under the KA.

5. CONCLUSION AND FUTURE WORK

Summarizing, we proposed a supporting system that recommends an effective and efficient path of learning objects for a given individual. To realize this system, a threefold architecture is needed—Domain model, Learner model and Adaptation Model. As an initiative step, we conducted a course content analysis, in which Labeled LDA was utilized to predict the knowledge coverage of a course. The result provided the positive indication that involving external explanatory texts on domain knowledge facilitates the prediction of the knowledge coverage of unknown course syllabi. However, the precision of the the current experiment needs further improvement in addressing texts semantically. Specifically, a bigram or trigram method is expected to perform better than the unigram method. In addition, separate nouns and noun-phrases may increase the precision. From a holistic perspective, we also need to consider the estimation of learner characteristics when constructing domain knowledge bases. For example, a framework of knowledge that connects knowledge itself with its learning outcomes may be instrumental in mapping learning objects to learners.

6. ACKNOWLEDGEMENTS

This work was supported by JSPS KAKENHI Grant Number 15K00423 and the Kayamori Foundation of Informational Science Advancement.

7. REFERENCES

[1] T. Barnes. Q-matrix Method: Mining Student Response Data for Knowledge. Technical report, 2005.

[2] Y. Belanger and J. Thornton. Bioelectricity: A Quantitative Approach Duke University's First MOOC. Report, 2013.

[3] R. J. C. Bose, O. Deshmukh, and B. Ravindra. Discovering Concept Maps from Textual Sources. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[4] P. Brusilovsky, J. Eklund, and E. Schwarz. Web-based education for all: a tool for development adaptive courseware. *Computer Networks and ISDN Systems*, 30(1-7):291–300, 1998.

[5] F. Gasparetti, C. Limongelli, and F. Sciarrone. Exploiting wikipedia for discovering prerequisite relationships among learning objects. In *Proceedings of International Conference on Information Technology Based Higher Education and Training*, pages 1–6, 2015.

[6] A. D. Ho, I. Chuang, J. Reich, C. A. Coleman, J. Whitehill, C. G. Northcutt, J. J. Williams, J. D. Hansen, G. Lopez, and R. Petersen. HarvardX and MITx: Two Years of Open Online Courses Fall 2012-Summer 2014. SSRN Scholarly Paper ID 2586847, Social Science Research Network, 2015.

[7] I. Kiyoshi et al. Investigation on the Educational Contents among Informational Science and Engineering Departments by Using Syllabus (Intermediate Report). Technical Report 6, Information Processing Society of Japan, 2010.

[8] A. Klašnja-Milićević, B. Vesin, M. Ivanović, and Z. Budimac. E-Learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Education*, 56(3):885–899, 2011.

[9] F. W. B. Li, R. W. H. Lau, and P. Dharmendran. An Adaptive Course Generation Framework. *Int. J. Distance Educ. Technol.*, 8(3):47–64, July 2010.

[10] N. Matsuda, T. Furukawa, N. Bier, and C. Faloutsos. Machine beats experts: Automatic discovery of skill models for data-driven online course refinement. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[11] A. Paramythis and S. Loidl-Reisinger. Adaptive Learning Environments and eLearning Standards. *ELECTRONIC JOURNAL OF ELEARNING, EJEL: VOL 2. ISSUE*, 2:181–194, 2004.

[12] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: A Supervised Topic Model for Credit Attribution in Multi-labeled Corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256. Association for Computational Linguistics, 2009.

[13] T. Sekiya, Y. Matsuda, and K. Yamaguchi. Curriculum analysis of CS departments based on CS2013 by simplified, supervised LDA. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 330–339, 2015.

[14] I. C. Society. Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. Technical report, ACM, 2013. 999133.

[15] S. Sosnovsky and P. Brusilovsky. Evaluation of topic-based adaptation and student modeling in QuizGuide. *User Modeling and User-Adapted Interaction*, 25(4):371–424, 2015.

[16] J. C. R. Tseng, H.-C. Chu, G.-J. Hwang, and C.-C. Tsai. Development of an adaptive learning system with two sources of personalization information. *Computers & Education*, 51(2):776–786, 2008.

[17] S.-S. Tseng, P.-C. Sue, J.-M. Su, J.-F. Weng, and W.-N. Tsai. A new approach for constructing the concept map. *Computers & Education*, 49(3):691–707, 2007.

[18] A. Vuong, T. Nixon, and B. Towle. A method for finding prerequisites within a curriculum. In *Proceedings of the 4th International Conference on Educational Data Mining*, 2011.

[19] J. Řihák, R. Pelánek, and J. Nižnan. Student Models for Prior Knowledge Estimation. In *Proceedings of the 8th International Conference on Educational Data Mining*, pages 109–116, 2015.

Student Emotion, Co-occurrence, and Dropout in a MOOC Context

John Dillon
Univ. of Notre Dame
jdillon5@nd.edu

Nigel Bosch
Univ. of Notre Dame
pbosch1@nd.edu

Malolan Chetlur
IBM Research, India
mchetlur@in.ibm.com

Nirandika Wanigasekara
IBM Research, India
nwaniga4@in.ibm.com

G. Alex Ambrose
Univ. of Notre Dame
gambrose@nd.edu

Bikram Sengupta
IBM Research, India
bsengupt@in.ibm.com

Sidney K. D'Mello
Univ. of Notre Dame
sdmello@nd.edu

ABSTRACT

This paper discusses self-reported emotions experienced by students in a Massive Open Online Course (MOOC) learning context. Emotions have been previously shown to be related to learning in classrooms and laboratory studies and have even been leveraged to improve learning. In this study, frequently occurring discrete emotions as well as frequently, co-occurring pairs of emotions were analyzed during learning with a MOOC. Both discrete and co-occurring emotions were related to students dropping out of the course, illustrating the importance of student emotion in a MOOC context.

Keywords

MOOC; affective computing; course completion.

1. INTRODUCTION

Emotion is one of the key aspects of the learning process [9,22]. It influences learning in a variety of ways [12], both positively (e.g., when a student feels engaged [19]) and negatively (e.g., during boredom [6,19]). These connections between emotion and cognition can be leveraged to improve learning [10]. For example, a dialog-based, intelligent tutor that adjusts its dialog to address negative emotions can improve learning for low-knowledge students [11]. Indeed, the relationship between emotion and learning has been researched in a variety of digital learning contexts in both laboratory studies and classroom studies [1,5,9]. There are, however, additional learning contexts in which the relationship between emotion and learning is less clear. In this study we focus on the role of emotion as it relates to student dropout in the context of a Massive Open Online Course (MOOC).

MOOCs are an online learning context that has recently become popular worldwide [18]. MOOCs provide education access to large groups of people, many of whom are often non-traditional students. Little is known about the relationship between emotions and learning in a MOOC context. Some initial work toward examining emotion in MOOCs indicated that some emotions were related to dropout [13]. However, these results were derived from retrospective reports of emotion after a course rather than reports in the moment, i.e., *during* the course. Similarly, studies have used MOOC discussion forums and clickstream data to infer student emotions such as *Confusion* and *Frustration* based on researchers' judgments of how these emotions are manifested [16,27], but there was no measurement of the emotions from the students themselves.

The current paper expands on this limited research, addressing key open questions about student emotions gathered from self-reports at different points in a MOOC. We explore a range of emotions, including *Anger*, *Boredom*, *Confusion*, *Contentment*, *Disappointment*, *Enjoyment*, *Frustration*, *Hope*, *Hopelessness*, *Isolation*, *Pride*, *Relief*, *Sadness*, and *Shame*, while also focusing on the relationship between *Anxiety* and learning statistics (the focus of the MOOC in this study) [8,17].

We also consider the possibility of co-occurring emotions. Decades ago, Izard et al. [14] considered the possibility that certain emotions may be experienced in concert with other emotions, rather than individually. Experimental research has shown this to be the case in some situations, for example with induced emotions and even with emotions experienced during everyday life [3,21]. In the context of learning, Bosch and D'Mello [4] studied novice programmers' emotions and found *Confusion* co-occurred with *Frustration*, while *Curiosity* co-occurred with *Engagement*. The degree of co-occurrence of *Curiosity* and *Engagement* was positively correlated with learning (r = .226) after accounting for individual occurrences, thereby highlighting the importance of examining co-occurring emotions.

In addition to tracking the incidence of emotions and co-occurrence pairs, we also consider how emotions are related to key educational outcomes. Early studies of MOOC data and student behavior [7,26], have often focused on "dropout" as both a problem and a key outcome. Recently, some have questioned the validity of dropout as a metric of outcome assessment [13]. However, Yang et al. [26] have noted, for instance, that the very low completion rates of MOOCs should signal some concern. Researchers have used log data to predict student dropout [15,23] as part of a larger effort aimed at better understanding student dropout from MOOCs and, in turn, improving the MOOC learning experience to reduce dropout. Here, we consider the relationship between students' self-reported emotions and course dropout.

To our knowledge, this is the first study to measure a range of self-reported student emotions in a MOOC context. We believe that the opportunity to study student emotion with large courses in the wild offers a valuable addition to previous work that has focused more on laboratory settings or traditional classroom environments. We address three related questions in this research:

- Q1. What emotions do students experience in a MOOC?
- Q2. Which emotion pairs co-occur more than chance?
- Q3. How do individual and co-occurring emotions relate to dropout?

2. METHOD AND COURSE SETUP

"I Heart Stats" was an introductory Statistics MOOC offered by a university in the Midwestern United States. One goal of the course was to alleviate student anxiety towards statistics. In this regard it was a prime opportunity to analyze student affect in a MOOC setting, while also providing an opportunity to study student affect at scale in the wild.

This MOOC contained eight modules covering topics ranging from levels of measurement to ANOVA. Modules were designed to be completed in sequential order. Nevertheless, all modules were released to students at the same time, so students were free to complete the modules at their own pace and in whatever order they desired.

We used a "Pick-Two" list of 15 discrete emotions (Figure 1) to measure student affect. In addition to the typical set of learning-centered affective states like *Confusion* and *Boredom* [9], the list included several additional emotions, such as *Enjoyment*, *Pride*, *Isolation*, *Hope*, and *Shame*. These emotions were, in part, selected from Pekrun's description of academic emotions [20]. One limitation of this emotion list was that *Neutral* was not included. Students were prompted to report emotions at the start of even-numbered modules (0, 2, 4, 6) as well as at the end of module 8 (last module). We only collected affect reports on every other module to minimize intrusion.

Of the 24,279 students from 183 different countries enrolled in the course, 3,591 students reported exactly two emotions on at least one module. These 3,591 students constituted the sample in this study. Students were able to report greater or fewer than two emotions, but because we were interested in co-occurrence, we excluded responses that did not consist of exactly two emotions.

Figure 1. Emotion reporting list

In addition to five "course-level" affect surveys, in which students reported their emotions in relation to the course as a whole, we also included seven "content-level" surveys. These content-level surveys were spread throughout the course and prompted students to report their emotions in response to different video lectures and problem sets. These are two common content-delivery methods for MOOCs, thereby providing a preliminary understanding of student affect when completing these two activities.

3. RESULTS

We used both the course-level and content-level students self-reported emotions to answer our research questions (see Introduction).

Q1. What emotions do students experience in a MOOC?

Figure 2 presents the aggregated proportions of each reported emotion across all five course-level surveys. We note that *Hope* and *Enjoyment* were the most frequently reported emotions. Other frequently reported emotions were *Contentment, Anxiety,* and *Pride,* while *Shame, Disappointment, Isolation, Anger*, and *Sadness* were rarely reported.

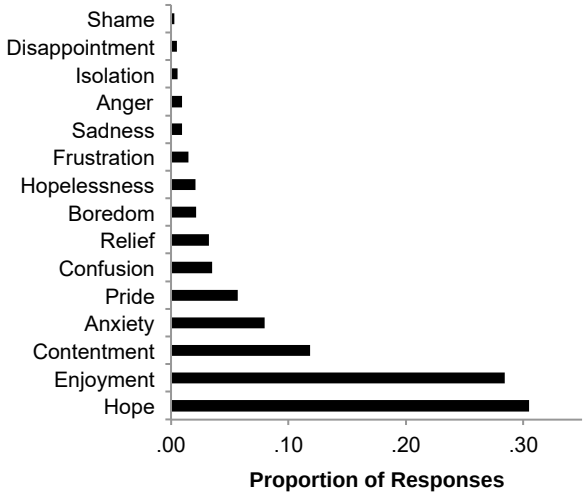


Figure 2. Proportions of self-reported emotions

These results differ from the recent D'Mello meta-analysis [9], where the studies rarely included emotions such as *Hope*, *Enjoyment*, and *Contentment.* However, the focus there was on short one-on-one interactions during learning with technology. A different set of emotions appear to be playing a critical role in the MOOC context, so context clearly matters. It is, however, difficult to separate context differences from measurement differences in the present study.

In addition to the course-level emotion surveys, we also included content-level affect surveys to assess self-reported emotion in relation to specific segments of content that may elicit different emotional responses. We selected 4 content-level affect surveys to highlight different affective states across video and problem set sections of content. Two of the activities were instruction videos and the other two were homework and practice problem sets. We excluded emotions that occurred in less than 1% of the responses for each specific activity. In addition, since all of the content for this course was released at the same time, we use log timestamps to ensure that: 1) Students engaged with the activity, 2) Students answered the activity-specific affect question *after* their engagement with the activity, and 3) Students did not take more than 1 hour following the last activity log to complete the emotion survey.

Figure 3 presents the emotion proportion distributions for four learning activities. The results indicated that unlike the course-level emotion reports, *Enjoyment* was more frequent than *Hope*. Further, while *Anxiety* was the fourth most commonly reported

emotion at a course-level, it was far less prominent at the content level.

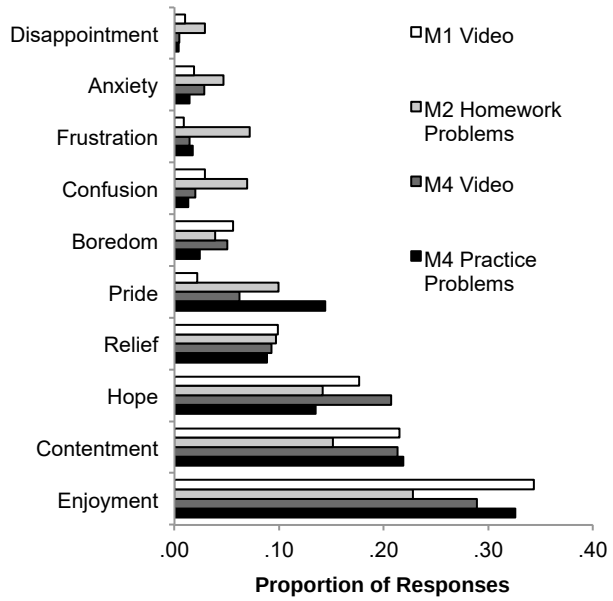


Figure 3. Proportion of emotion self-reports by activity type

We also note that the content-level emotions varied with regard to certain activities. For instance, *Pride* was reported nearly 10 times more frequently in response to Module 4 Practice Problems than in Module 1 Video. *Frustration, Confusion,* and *Anxiety* were quite prominent during Module 2 Homework Problem compared with Module 4 Video. *Relief*, on the other hand, did not fluctuate substantially among these four content-level reports. *Hope* was more frequently reported in both of the video activities, while *Pride* was more frequently reported in the problem sets. Further research is needed to determine if indeed students expressed *Pride* more frequently in contexts of achievement such as completing a problem set. We would also need to consider a larger set of activities to establish if certain emotions occur more frequently and significantly among certain genres of content.

These course-level affect surveys highlight that students experience different emotions during different types of content in a MOOC. If MOOCs are able identify the prominent emotions associated with various types of content such as videos and problem sets, then instructors and course designers can provide appropriate support to learners when needed.

Q2. Which emotion pairs co-occur more than chance?

Bosch and D'Mello [4] investigated co-occurrence of emotions in a computerized learning environment. In their study, they employed a retrospective judgment protocol without any interruptions during the learning session. They determined which co-occurring emotions occurred more than chance by computing Lift scores [24] for each emotion pair. Lift is a technique from association rule learning that can be used to compare the observed co-occurrence of emotions to the level expected by chance. Lift of a pair of emotions (X, Y) is defined as ratio of Pr(X and Y) to Pr(X)*Pr(Y).

We identified co-occurring course-level emotions as follows. First, we only considered responses with exactly two emotion reports. Second, we only considered affective states that occurred at least 1% of the time. This restricted our analysis to *Anxiety*, *Boredom*, *Confusion*, *Contentment*, *Enjoyment*, *Frustration*, *Hope*, and *Pride*. Lift scores were calculated for all pairwise combinations of the above emotions. We used random sampling without replacement (1,000 iterations) and a sample size of 3,000 to compute 95% bootstrapped confidence intervals for the Lift scores. Lift scores above 1.0 with confidence intervals that do not overlap with 1.0 are considered to occur more frequently than chance.

We computed Lift scores for all 5 course-level affect reports. There were 92 distinct co-occurring emotions and a total of 5,189 emotion pairs as reported by 3,591 learners. The results are shown in Table 1. We note that only 5 out of the possible 92 emotion combinations co-occurred at levels above chance and these mainly involved the learning-centered affective states of *Confusion*, *Frustration*, *Boredom*, and *Anxiety*. The *Confusion* + *Frustration* pair had the highest Lift score, which is consistent with [4] despite considerable differences in the temporal resolution of the analyses. Somewhat surprising is the fact that *Boredom* co-occurred with both *Confusion* and *Frustration*, but this might be attributed to the coarse-grained nature of the emotion self-reports (e.g., *Boredom* could occur for some activities and *Confusion* for others within the same session).

Table 1. Lift of frequently co-occurring emotion combinations

Emotion Pair	Mean (SD)	Confidence Interval
Anxiety + Frustration	1.22 (0.17)	(1.21, 1.22)
Boredom + Confusion	1.06 (0.23)	(1.05, 1.06)
Boredom + Frustration	1.39 (0.43)	(1.39, 1.4)
Confusion + Frustration	3.22 (0.41)	(3.21, 3.23)

Q3. How do individual and co-occurring emotions relate to dropout?

We coded a student as having "dropped out" if he or she had no interaction events in the last module (Module 8). Table 2 presents partial Spearman's *rho* between dropout and course-level discrete emotions that comprised at least 1% of the data and corresponding exceeding chance. We partialled out the number of emotion reports per student in order to control for the steep rate of attrition and subsequent dropout bias in our data.

The results indicated that *Anxiety*, *Confusion*, and *Frustration* were significantly positively correlated with dropout, which is what we would expect. It was surprising, however, that *Hope* was also positively correlated with dropout, suggesting that these hopeful students might have become disillusioned by the MOOC. *Relief* was weakly negatively related to dropout, albeit non-significantly.

Table 2. Partial correlations between affect reports and dropout

Emotion/ Combination	*rho*	*p*
Anxiety	**.155**	**.000**
Boredom	.004	.954
Confusion	**.122**	**.019**
Contentment	-.035	.243
Enjoyment	-.028	.184
Frustration	**.251**	**.003**
Hope	**.046**	**.018**
Pride	.034	.476
Relief	-.081	.145
Anxiety + Frustration	.107	.458
Boredom + Confusion	-.088	.684
Boredom + Frustration	-.018	.956
Confusion + Frustration	.177	.263

The most valuable payoffs of this study for learning scientists and MOOC designers are the positive, though weak, correlations between *Frustration*, *Anxiety*, *Confusion* and dropout. The next step is to identify the causes or partial causes of those negative emotions. For example, students reported three times more *Frustration* in Module 2 Homework Problems than in other selected activities, suggesting that the homework problems in this module might need deeper consideration.

4. DISCUSSION

We recorded student affect in a MOOC setting and analyzed them with respect to both individual emotions and co-occurring pairs. This study marks the first large-scale analysis of self-reported emotion in a MOOC context. We found that students experience a rather diverse set of emotions while completing a MOOC in comparison with previous work that has focused on lab- or in-class learning. Particularly interesting was the finding that *Hope*, *Enjoyment*, and *Contentment* were the most frequently reported emotions in the MOOC context, given that they are rare in shorter learning sessions studied in previous work [9].

We also found that some emotions fluctuate depending on MOOC content. This is an especially valuable finding for both instructional designers and researchers. From a learning design perspective, if we know how students are affectively reacting to different types of content, we can adjust the course materials accordingly.

Our findings also contribute to the dropout problem in MOOCs. Despite researchers capacity to predict dropout [25,26], we still lack a robust understanding of student dropout. We identified specific emotions and emotion combinations that correlate with student dropout, yielding an affective perspective to the dropout problem.

5. LIMITATIONS AND FUTURE WORK

There are several limitations with this exploratory study. First, the content was released to students all at once, so they could complete the course in any order they desired. This limits the feasibility of temporal analysis of the data. Second, since this study was based on a live course, we could not ask students to self-report their affective states as frequently as in a lab setting. This limits use of the data for more fine-grained sequential analyses.

Our analyses also point to several opportunities for future work. One promising avenue is sensor-free affect detection for MOOCs [2]. It would be valuable to model student emotion based entirely on clickstream data provided by edX and other online learning platforms. This would allow for far more frequent affect measurement and more timely affect intervention. If, for instance, we know, based on log data, that a student is *Frustrated*, and we know that *Frustration* correlated with dropout, we can launch pedagogical scaffolds to help the student manage his or her *Frustration*.

A second opportunity for future work is to analyze changes in emotions across the time. There are many questions that can be asked along this front. How do emotions change over the duration of an activity, a session, or the entire course? What is the affective trajectory of a successful MOOC student? Further research is needed to map emotion trajectories over the duration of the course so that we better understand the relationships between emotions, their temporal dynamics, and educational outcomes.

6. ACKNOWLEDGMENTS

We would like to thank Crystal DeJaegher and Xiaojing Duan in the Office of Digital Learning at the University of Notre Dame for their assistance in the design of this MOOC and collection of these data. D'Mello was supported by the National Science Foundation (NSF) (DRL 1235958 and IIS 1523091). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

7. REFERENCES

1. Ivon Arroyo, David G. Cooper, Winslow Burleson, Beverly Park Woolf, Kasia Muldner, and Robert Christopherson. 2009. Emotion sensors go to school. *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, IOS Press, 17–24.
2. Ryan Baker, Sujith M. Gowda, Michael Wixon, et al. 2012. Towards sensor-free affect detection in cognitive tutor algebra. *Proceedings of the 5th International Conference on Educational Data Mining*, 126–133.
3. Lisa Feldman Barrett. 1998. Discrete emotions or dimensions? The role of valence focus and arousal focus. *Cognition & Emotion* 12, 4: 579–599.
4. Nigel Bosch and Sidney D'Mello. 2014. Co-occurring affective states in automated computer programming education. *Proceedings of the Workshop on AI-supported Education for Computer Science (AIEDCS) at the 12th International Conference on Intelligent Tutoring Systems*, 21–30.
5. Nigel Bosch, Sidney D'Mello, Ryan Baker, et al. 2015. Automatic detection of learning-centered affective states in the wild. *Proceedings of the 2015 International Conference on Intelligent User Interfaces (IUI 2015)*, New York, NY: ACM, 379–388.
6. Nigel Bosch, Sidney D'Mello, and Caitlin Mills. 2013. What emotions do novices experience during their first computer programming learning session? *Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED 2013)*, Berlin Heidelberg: Springer-Verlag, 11–20.
7. Lori Breslow, David E. Pritchard, Jennifer DeBoer, Glenda S. Stump, Andrew D. Ho, and Daniel T. Seaton. 2013.

Studying learning in the worldwide classroom: Research into edX's first MOOC. *Research & Practice in Assessment* 8: 13–25.
8. Peter K. H. Chew and Denise B. Dillon. 2014. Statistics anxiety update: Refining the construct and recommendations for a new research agenda. *Perspectives on Psychological Science* 9, 2: 196–208.
9. Sidney D'Mello. 2013. A selective meta-analysis on the relative incidence of discrete affective states during learning with technology. *Journal of Educational Psychology* 105, 4: 1082–1099.
10. Sidney D'Mello, Nathan Blanchard, Ryan Baker, Jaclyn Ocumpaugh, and Keith Brawner. 2014. I feel your pain: A selective review of affect-sensitive instructional strategies. In *Design Recommendations for Intelligent Tutoring Systems - Volume 2: Instructional Management*, Robert Sottilare, Art Graesser, Xiangen Hu and Benjamin Goldberg (eds.). 35–48.
11. Sidney D'Mello, Blair Lehman, and Art Graesser. 2011. A motivationally supportive affect-sensitive AutoTutor. In *New Perspectives on Affect and Learning Technologies*, Rafael A. Calvo and Sidney K. D'Mello (eds.). Springer New York, 113–126.
12. K. Fiedler and S. Beier. 2014. Affect and cognitive processes in educational contexts. *International handbook of emotions in education*: 36–56.
13. Christian Gütl, Rocael Hernández Rizzardini, Vanessa Chang, and Miguel Morales. 2014. Attrition in MOOC: Lessons learned from drop-out students. In *Learning Technology for Education in Cloud. MOOC and Big Data*, Lorna Uden, Jane Sinclair, Yu-Hui Tao and Dario Liberona (eds.). Springer International Publishing, 37–48.
14. Carroll E. Izard and Edmund S. Bartlett. 1972. *Patterns of emotions: A new analysis of anxiety and depression*. Academic Press, Oxford, England.
15. Suhang Jiang, Mark Warschauer, Adrienne E. Williams, Diane O'Dowd, and Katerina Schenke. 2014. Predicting MOOC performance with week 1 behavior. *Proceedings of the 7th International Conference on Educational Data Mining*, 273–275.
16. Derick Leony, Pedro J. Muñoz-Merino, José A. Ruipérez-Valiente, Abelardo Pardo, and Carlos Delgado Kloos. 2015. Detection and evaluation of emotions in massive open online courses. *Journal of Universal Computer Science* 21, 5: 638–655.
17. Anthony J. Onwuegbuzie, Denise Da Ros, and Joseph M. Ryan. 1997. The components of statistics anxiety: A phenomenological study. *Focus on Learning Problems in Mathematics* 19, 4: 11–35.
18. Laura Pappano. 2012. The year of the MOOC. *The New York Times*.
19. Zachary A. Pardos, Ryan S. J. D. Baker, Maria O. C. Z. San Pedro, Sujith M. Gowda, and Supreeth M. Gowda. 2013. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, ACM, 117–124.
20. Reinhard Pekrun, Thomas Goetz, Wolfram Titz, and Raymond P. Perry. 2002. Academic emotions in students' self-regulated learning and achievement: A program of qualitative and quantitative research. *Educational Psychologist* 37, 2: 91–105.
21. Janet Polivy. 1981. On the induction of emotion in the laboratory: Discrete moods or multiple affect states? *Journal of Personality and Social Psychology* 41, 4: 803–817.
22. Paul Schutz and Reinhard Pekrun (eds.). 2007. *Emotion in Education*. Academic Press, San Diego, CA.
23. Tanmay Sinha, Patrick Jermann, Nan Li, and Pierre Dillenbourg. 2014. Your click decides your fate: Inferring information processing and attrition behavior from MOOC video clickstream interactions. *arXiv:1407.7131 [cs]*.
24. Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. 2002. Selecting the Right Interestingness Measure for Association Patterns. *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 32–41.
25. Colin Taylor, Kalyan Veeramachaneni, and Una-May O'Reilly. 2014. Likely to stop? Predicting stopout in massive open online courses. *arXiv:1408.3382 [cs]*.
26. Diyi Yang, Tanmay Sinha, David Adamson, and Carolyn Penstein Rose. 2013. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. *Proceedings of the 2013 NIPS Data-driven education workshop*, 1–8.
27. Diyi Yang, Miaomiao Wen, Iris Howley, Robert Kraut, and Carolyn Rose. 2015. Exploring the effect of confusion in discussion forums of massive open online courses. *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*, ACM, 121–130.

Semi-Markov model for simulating MOOC students

Louis Faucon, Łukasz Kidziński, Pierre Dillenbourg
Computer Human Interaction in Learning and Instruction
École Polytechnique Fédérale de Lausanne
{louis.faucon,lukasz.kidzninski,pierre.dillenbourg}@epfl.ch

ABSTRACT

Large-scale experiments are often expensive and time consuming. Although Massive Online Open Courses (MOOCs) provide a solid and consistent framework for learning analytics, MOOC practitioners are still reluctant to risk resources in experiments. In this study, we suggest a methodology for simulating MOOC students, which allow estimation of distributions, before implementing a large-scale experiment.

To this end, we employ generative models to draw independent samples of artificial students in Monte Carlo simulations. We use Semi-Markov Chains for modeling student's activities and Expectation-Maximization algorithm for fitting the model. From the fitted model, we generate simulated students whose processes of weekly activities are similar to these of the real students.

Keywords

MOOCs; simulation of students; generative models; Expectation-Maximization; Semi-Markov chains; Bayesian statistics

1. INTRODUCTION

Vast amounts of data which we gather and analyse in modern learning environments allow us to build models of unprecedented scale and accuracy. This phenomenon, in parallel with developments in computer science, gave rise to new possibilities of inference from educational environments. In particular, the growing field of Simulated Learners [8, 11, 14] provides us with tools for inference from educational simulations.

Inference from any simulations is bounded by the predefined level of abstraction of the analysis. In the context of Massive Online Open Courses (MOOCs), on one hand as an educational institution we have access to only a handful of MOOCs, on another hand, we have data as granular as student's clickstream in a video player. We are therefore obliged to model granularity robustly, depending on the availability of the data. We argue that understanding the properties of the statistical methodology at hand is crucial for successful inference.

We propose a probabilistic model, based on extended version of Markov Chains, called semi-Markov Chains. In the model, we can balance the complexity of the structure and the number of parameters to estimate by cross-validating its parameters. We present an algorithm for fitting the model as well as illustrative examples of the fit on a set of MOOCs.

The contributions of this paper are threefold. First, **we investigate to what extent Semi-Markov chains can be used to describe behavioural patterns of students (RQ1)**. Second, since our model implicitly divides users into clusters, **we analyse if these clusters are interpretable (RQ2)**. Third, **we analyse how these models can be used to infer distributions of events (RQ3)**.

2. RELATED WORK

Modeling students is a key concept in learning analytics and educational research in general. Researchers build models predicting motivation and cognition, based on student's goals [19] or they predict goals by motivational traits [7]. Large datasets allow researchers to find predictive power of seemingly slightly related signals like the length of pauses in a video [12] or potentially noisy signals like head movement in the classroom [16].

2.1 Generative models in MOOCs

All the aforementioned models are focused on prediction and belong to the class of so-called discriminative models. In this study, we suggest a generative model, which allow us not only to predict, but also to generate observations from the estimated distribution. These models capture the probability structure of input variables and the flow of the processes. Several generative models in MOOCs have been applied, e.g. to forums [3].

Among many generative models that can be encountered in educational research, Markov models were employed for visualization [5], for modeling engagement [17] and for modeling students retention [1].

2.2 Simulated learner

The area of simulating students' behaviour lays on the intersection of cognitive science and artificial intelligence. Examples of applications of simulation of students can be found even outside computer science, where the teacher simulates student's response in order to self-improve instructional skills [18]. An acknowledged example of the usage of simulating humans [9] for education deals with simulations of patients behaviour for training medicine students.

Emergence of Internet and new data storage techniques allow re-

searchers to collect and analyse massive amounts of information about the users. Researchers employ simulations for clustering students [13]. For a review of earlier techniques we refer to [2]. We motivate our methodology by the advancements of user modeling in web context [4], as we find this environment conceptually close to the environment of a MOOC.

3. GENERAL FRAMEWORK

3.1 Dataset

From our internal MOOC database, aggregating data from Coursera and edX, we extracted events for 61 EPFL courses. The raw data contained approximately 23 million events for 500,000 students, arranged in tuples: <`StudentID, CourseID, EventType, Timestamp`>. The *EventType* describes the type of an activity and takes one of four possible values presented in Table 1. We choose these events as the most discriminative actions from the key areas: learning, validation and community engagement. Note that our modelling technique can be easily extended to cover other types of events.

Abbreviation	Description	Proportion
`VideoPlay`	watching a video	51%
`Submission`	submitting an assignment	33%
`ForumView`	visiting the forum	15%
`ForumPost`	posting on the forum	1%

Table 1: Distribution of events in the dataset.

For the analysis we developed our own Python implementation of the algorithm fitting the model[1]. In Section 5 we explain the algorithm in detail. Since 23 million events can still fit in memory of a single computer, we did not require a specific computing architecture to perform the analysis. However, given the considerable size of the dataset, the algorithm takes several minutes to run.

3.2 Definitions

We start with a general framework, in which student's activity in any MOOC can be very precisely described. Next, we elevate abstraction of the model by adding assumptions simplifying the analysis. Our goal is to introduce a model whose complexity can be adapted to the structure of a course and the amount of available data.

We consider a model in which students behaviour is described in a sequential manner by the type of activity they perform and the time they wait between two sessions. Furthermore, as most of the students perform at most 1 MOOC session per day, we choose a daily granularity of actions.

A sequence of student's daily activity is described as a list of 'active events' (`VideoPlay`, `Submission`, `ForumView` and `ForumPost`) followed by a 'end of the day event' (`EndOfDay`) or only a `EndOfDay` in the case the student did not perform any activity the given day. The formal definition of the model is following:

The set of all students $\mathscr{S}$: We use the symbol $s \in \mathscr{S}$ to designate an individual student.

The set A of all types of activities: For this study we chose a set of four types of events: { `VideoPlay`, `Submission`, `ForumView`, `ForumPost` }. We add to this set one special type of event, `EndOfDay`. This event corresponds to the end of interactions with MOOCs on a given day. We use the symbol $a \in \mathbf{A}$ to designate any type of activity. One can extend the set of activities to other events if needed for certain application.

Note that we do not specify the regular 'end of a course' event, since we only model the behaviour within the limited time-frame of a course and we treat the last day of the course as the last day of the process. Therefore, each student who went through the whole course without dropping out has just a `EndOfDay` event on the last day of the course. Number of `EndOfDay` events is therefore equal to the number of days of the course.

The random sequential variable $\mathbf{X}_1^{(s)}, \mathbf{X}_2^{(s)}, ..., \mathbf{X}_n^{(s)}$ represents the sequence of activities of one student s. Each $\mathbf{X}_i^{(s)} \in \mathbf{A}$ and the sequence stops after an `EndOfDay` when the student reaches the end of the course. We denote the length of the sequence for a student s as $n^{(s)}$. The observation of one student activity along one MOOC is thus a **realization** of the random sequence $\mathbf{X}$.

The probability distribution P: In general, for each student $s \in \mathscr{S}$ we can model the i-th event $\mathbf{X}_i^{(s)}$ with a probability distribution

$$\mathbf{P}^{(s)}(\mathbf{X}_i^{(s)} = a \mid \mathbf{X}_{i-1}^{(s)}, , \mathbf{X}_{i-2}^{(s)}, ..., \mathbf{X}_1^{(s)}, \mathbf{C}_s),$$

where $a \in \mathbf{A}$, $\mathbf{X}_1^{(s)}, ..., \mathbf{X}_{i-1}^{(s)}$ are the previous events of that student and $\mathbf{C}_s$ are personal characteristics of the student.

This distribution represents the student's behaviour profile and allows to generate typical sequences of activities. Our main objective is to model this distribution as accurately as possible, given the limited information. The accurate distribution would allow us to draw samples of students.

3.3 Assumptions

As discussed in the previous section, assessing $\mathbf{P}^{(s)}$ is unfeasible due to dependence on too many events in the past and due to the lack of information on personal student features. In order to fit a probabilistic model we need to relax these dependencies. We introduce following assumptions:

- **A1** Students' behaviours fit into a small number of natural categories of behaviour.
- **A2** The type of activity depends only on his previous activity and not on old past activities.

Assumption **A1** maps the space of all possible students' characteristics into a limited number of categories, which are much easier to attribute. Many studies on MOOCs explicitly classify students into a small number of categories [10], students are divided between 'Viewers' who only watch videos, 'Forum Actives' who share with their peers in the MOOC discussion forum and 'Completers' who succeed in the assignments. As we present in the next section, our method is based on unsupervised clustering, where groups emerge in the way optimal in terms of maximum likelihood of the model.

Assumption **A2** we impose that only the last activity has an impact on the current activity. This assumption is more constraining, but since the complexity of history grows exponentially with the number of steps and, in order to be able to estimate parameters, we have to

[1]Our implementation is available under https://github.com/lfaucon/edm2016-mooc-simulator

reduce the search space. This simplification is usually called the 'Markov assumption'.

Apart from technical assumptions required for Markov Models, we impose other assumptions for convenience. First, we do not consider length of events, so the `VideoPlay` event is only the moment when a student starts watching a video. Second, if the series of events happens during midnight, still an event `EndOfDay` is added to the sequence.

4. PROBABILISTIC MODELING

4.1 Soft clustering

In Section 3 we proposed a simplified framework, in which we assume that there are only a few different possible classes of students **(A1)**. We enumerate clusters $1,2,...,K$. For each student $s \in \mathscr{S}$ we introduce a probability distribution $\mu_k^{(s)}$ which describes probability that the student belongs to the behaviour classes k, for $k \in \{1,2,...,K\}$.

This technique is often referred to as *soft clustering*, *weighted clustering* or *fuzzy clustering* [15]. Instead of discret cluster assignment, as for example in K-means, we obtain for each student a probability distribution among the clusters. These probabilities can be intuitively seen as our certainty that the student belongs to a given cluster.

4.2 Semi-Markov Chain

Assumption **(A2)**, i.e. dependence only on the last state, allows us to model the process Markov Chains. Formally, in the definition of distribution of the next event we can drop dependence of the events which occurred before the current one, i.e. we identify

$$\mathbf{P}^{(s)}(\mathbf{X}_i^{(s)} \mid \mathbf{X}_1^{(s)},...,\mathbf{X}_{i-1}^{(s)}) = P^{(s)}(\mathbf{X}_i^{(s)} \mid \mathbf{X}_{i-1}^{(s)})$$

A preliminary analysis revealed an important weakness of using classic Markov Models in our context. A traditional Markov model considers that a student is equally likely to stop watching videos when they have watched one, as when they have already watched ten videos. In practice, students watch videos sequentially and Markov Model does not capture appropriately the number of events in the sequence.

To remedy this issue we employed Semi-Markov Models (also called Markov Renewal Processes). The key feature of this model is that it allows to replace the self-loops (transitions from one event type to itself) in the Markov Chain, by a probability distribution of the number of repetition of a given state.

In Semi-Markov Models, we still need to choose a parametric distribution, but we have more freedom than in traditional Markov Chain. Markov Chain implicitly assumes that probability of staying in the same state is the largest for 1 step and decreases with number of steps. However, we would expect that 1 is not the most probable number of repetition at least for a particular group of students. This phenomenon can be captured by, for example, Poisson distribution, which proved to be more accurate in our preliminary analysis. Thus, for an event $a \in A$ and a class k we model the number of repeated events R_a^k by

$$\mathscr{P}(R_a^k = r) = \frac{e^{-\lambda_a^k}\left(\lambda_a^k\right)^r}{r!}$$

where r is the number of repetitions and λ_a^k is the average number of repetition and needs to be estimated from the data for each k and a.

To illustrate that the Poisson distribution improves the model, let us consider an example. Suppose we expect that some group of students connects to a MOOC twice a week, with approximately three days interval between connections. In that case, the average number of repetitions of the `EndOfDay` event is 3. Simple Markov Model, accurately models the average to be 3 but implicitly assumes that the majority of students gets only 1 repetition. Semi-Markov model with Poisson distribution also gives the average equal to 3 and the distribution is concentrated around 3.

5. FITTING THE MODEL

5.1 Algorithm

The Expectation-Maximisation (EM) algorithm has been introduced in 1977 in [6]. The goal of this iterative technique is to compute the parameters that maximize the likelihood of a given probabilistic model. The EM algorithm has been proven to converge at least to a local minimum. This minimum depends on the initialization point, thus multiple runs with different random initialisations are often used in practice in order to increase the chances of finding the global minimum.

In this study we use the EM algorithm for unsupervised learning. Neither the parameters of the latent classes nor the repartition of the students are known at the beginning and the algorithm has to estimate both quantities at once. In our settings, we define for each $k \in \{1,2,...,K\}$ and states a and b:

- $p_{b->a}^{(k)}$, the probability that a student with the behaviour profile k performs the activity a after the activity b:

$$p_{b->a}^{(k)} = \mathbf{P}(\mathbf{X}_i = a \mid \mathbf{X}_{i-1} = b)$$

- $\lambda_a^{(k)}$, the average number of repetitions of an event a from a student of profile k.

- $\mu_k^{(s)}$, the probability that a student s belongs to the profile k.

We can thus compute the likelihood of the observed sequence, as a function of cluster repartition and parameters of Markov Chains by

$$likelihood = \prod_{s \in \mathscr{S}} [\sum_{k=1}^{K} \mu_k^{(s)} \prod_{(a,b,r) \in \mathbf{T}_s} p_{b->a}^{(k)} \mathscr{P}_{\lambda_a^{(k)}}(r)], \qquad (1)$$

where $\mathbf{T}_s$ is the set of tuples $(a,b,r) \in \mathbf{A} \times \mathbf{A} \times \mathbf{N}$ corresponding to transitions from activity b to activity a with r repetitions of activity a. The goal of the algorithm is to find the parameters that maximize the likelihood.

In the first stage, the algorithm initialize randomly K profiles. Next, it iteratively improves the *likelihood*, by alternating two steps as described below. In each step it modifies the repartition or the Markov chain parameters.

Initialization: The initialization consists in choosing randomly either the $p_{b->a}^{(k)}$ and $\lambda_a^{(k)}$ or the $\mu_k^{(s)}$. In our algorithm, we start

with the $\mu_k^{(s)}$. This can be done by generating a random number k^* from 1 to K for each student s and by setting

$$\mu_k^{(s)} = \begin{cases} 1 & \text{if } k = k^* \\ 0 & \text{otherwise.} \end{cases}$$

Iterations: The iteration phase has two steps. First, we compute the optimal values for $p_{b->a}^{(k)}$ and $\lambda_a^{(k)}$ given that $\mu_k^{(s)}$ are fixed (equations (2) and (3)).

$$p_{b->a}^{(k)} = \frac{\sum_{s \in \mathscr{S}} \sum_{(a,b,\_) \in \mathbf{T}_s} \mu_k^{(s)}}{\sum_{s \in \mathscr{S}} \sum_{(\_,b,\_) \in \mathbf{T}_s} \mu_k^{(s)}} \tag{2}$$

$$\lambda_a^{(k)} = \frac{\sum_{s \in \mathscr{S}} \sum_{(a,\_,r) \in \mathbf{T}_s} r \mu_k^{(s)}}{\sum_{s \in \mathscr{S}} \sum_{(a,\_,\_) \in \mathbf{T}_s} \mu_k^{(s)}} \tag{3}$$

Next, we compute the new values of $\mu_k^{(s)}$ according to the new $p_{b->a}^{(k)}$ and $\lambda_a^{(k)}$ (equations (4)).

$$\mu_k^{(s)} = \frac{\prod_{(a,b,r) \in \mathbf{T}_s} p_{b->a}^{(k)} \mathscr{P}_{\lambda_a^{(k)}}(r)}{\sum_{c=1}^{K} \prod_{(a,b,r) \in \mathbf{T}_s} p_{b->a}^{(c)} \mathscr{P}_{\lambda_a^{(c)}}(r)} \tag{4}$$

Intuitively, the in the first step we compute the parameters of the latent classes given the repartition of the students and in the second step we recompute the repartition from the new classes parameters.

5.2 Example: Interpretation clusters (K=3)

Before we present the results for the choice of the number of clusters, in this section, we illustrate the behaviour of the algorithm and the model when the number of clusters is small ($K = 3$). Although in this case we may lose important variability among groups of students, small number of clusters allows us to visualise the Semi-Markov models and interpret each of the clusters.

The visualizations of the Semi-Markov models on Figure 1 can reveal general characteristics of students' behaviours. For example, Profiles 1 and 3 are in general less active as they have more `EndOfDay` events. On the contrary, Profile 3 has a very high average number of repetition on `VideoPlay` and considerable probability to go back to `EndOfDay` events. This means that students of this cluster are not fully engaged in all MOOC activities.

A more insightful way to analyse and interpret the differences is to generate sequences of events and compare the outcomes. We can compute the expected number of videos watched or the expected number of post on the forum directly from simulated sequences. Table 2 shows the average number of several types of events for 100 simulated students (average from 10000 simulations) over four weeks generated with the three Markov models from Figure 1. For

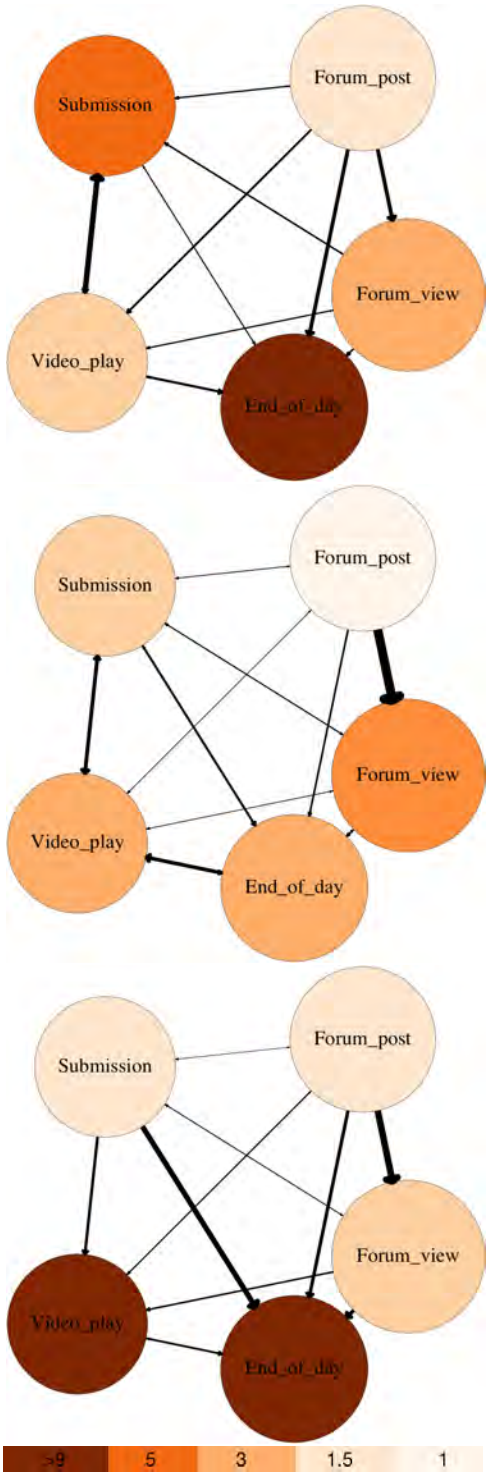


Figure 1: Three graphical representations of behaviour profiles extracted by the EM algorithm. From top to bottom: profiles 1, 2 and 3 (thickness: transition probability; color: average number of repetitions)

example, we can see that students of Profile 1 participate in the collaborative activities of the MOOC more rarely, but engage in the assignments more than in watching the videos. This might indicate

that they already have a good understanding of the content of the course and do not need to spend more time on studying. To fully investigate this hypothesis, further analysis should be conducted.

Profiles	1	2	3
Watched Videos	1060	3133	2363
Submissions	1535	2423	442
Forum Visits	68	1711	255
Forum posts	3	96	15

Table 2: Average number of events for 100 students over the first four weeks of the MOOC

5.3 Choice of the parameter K

A common challenge of unsupervised learning and fitting a probabilistic model is finding the correct number of classes. In our case, the similarity of the algorithm with other clustering techniques such as the K-means leads to the "elbow heuristic", often used in practice. The idea is to choose the number of clusters large enough to explain a large part of the variability, but such that a greater number of clusters would not explain substantially more.

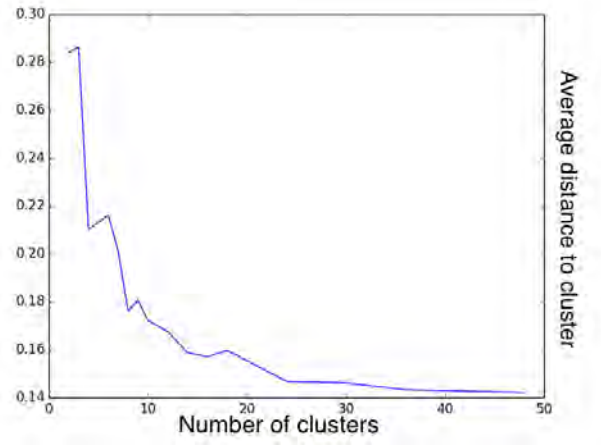


Figure 2: Average distance of students from their model for different number of classes

In order to confirm the result of this first measure of quality, we designed another measure described in the equation (5). The goal is to quantify how the students sequences diverge from their attributed cluster. In the equation, $|A|$ is the cardinality of the set of possible activities, $p_s(a)$ is the probability of finding the activity a if we take uniformly at random an activity of student s and $p_k(a)$ is the probability of finding the activity a if we take uniformly at random an activity from a sequence generated by the class k.

$$d^2(s,k) = \frac{1}{|A|} \sum_{a \in \mathbf{A}} (p_s(a) - p_k(a))^2 \tag{5}$$

This distance measure shows an elbow shape for the same values of K between 10 and 15 as it can be seen on Figure 2. We conclude that MOOC students from our dataset can be meaningfully clustered into $10-15$ different classes.

6. SIMULATIONS

With a model fitted with the EM algorithm at hand, the algorithm repartitioned students and chose parameters of a Semi-Markov Chain for each of the clusters. Since both the repartition and the Semi-Markov Chains are generative, we can draw samples from the fitted distribution, i.e. we can simulate the students. We run the simulations and show a possible way to measure the validity of the results.

To validate potential value of simulations, we first propose a simple accuracy measure. In equation (6), $P_{real}(|a| > n)$ represents the probability that a student performs more than n events of type a during the time of the MOOC. $|a|$ is the count of events of type a. $P_{sim}(|a| > n)$ represents the same probability but for a simulated student. In the measure we chose the value $N = 50$ because it covers most of the variability in the students activity sequences and is not too large as still 19% of the students have an activity with more than 50 repetitions.

$$MSE = \frac{1}{(|A|-1)*N} \sum_{a \in \mathbf{A}} \sum_{n<N} (P_{real}(|a| > n) - P_{sim}(|a| > n))^2 \tag{6}$$

In order to prove the correctness of the modeling method, we divided our dataset into a training set and a test set for validating the results. The first step is to run the algorithm on the training set with several parameter K and then, use the computed parameters to simulate a new population of students and finally compare this population with the students from the testing set. In Figure 3 we can see that the fit does not improve much after $K = 15$, because too high number of clusters makes the algorithm learn mostly the noise from the random actions of the students instead of their real intrinsic behavioural patterns.

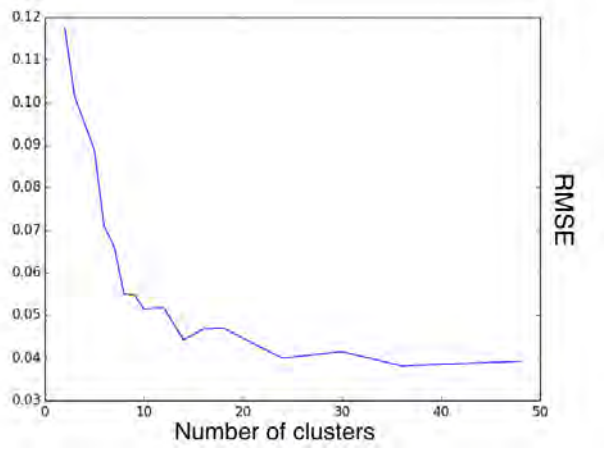


Figure 3: Measure of accuracy of a simulation for different number of classes

The small error proves that the distribution obtained from simulations is close to the original distribution. This implies that the model properly trained on small sample of students or on just few first events, can be extrapolated by simulation to further events or larger samples.

In an experimental setup, simulations with varying initial conditions of the model (e.x. probabilities of transitions) can give us distributions of events at the later state. Knowing probability distributions of the results of two conditions allows to estimate sample sizes needed for finding statistical evidence of the investigated effect.

7. DISCUSSION

In Section 5 we showed that Semi-Markov chains can be successfully applied to describe behavioural patterns of students **(RQ1)**. In Section 5.2, a simple study with reduced number of clusters prove their potential interpretability **(RQ2)**. In Section 6, we discuss how these models can be used to infer distributions of events **(RQ3)**.

Our method has two main limitations. They can be further relaxed with additional data or with incorporation of domain knowledge.

The Homogeneity of the Markov process: The Markov assumption was introduced for reducing the number of parameters of our model. It is a strong simplification, which entails some drawbacks. This assumption implicitly requires that student behave with exactly the same transition matrix during the whole course. The motivation to keep learning should increase when getting closer to the end of the course and thus the dropout rate decreases, which cannot be capture by our method. A good way to overcome this weakness is to use inhomogeneous Markov models with transitions probabilities that are functions of time.

Differences between courses: The quality of the videos, the level of difficulty of the assignments or the discussion topics in the forums are all factors that can greatly influence the behaviour of a student. None of these were included in our model. We hypothesize that adding external annotations that would impact the transition probabilities of our Markov models could help solve this problem. As for now, our model can be used to compare courses. For example, if we run the algorithm on two MOOCs and realise that the Video Watchers of one course have a lower engagement, that shows a lower quality of video content while differences for the Forum Follower may reveal differences on the quality of the Forum discussions.

8. REFERENCES

[1] Girish Balakrishnan and Derrick Coetzee. Predicting student retention in massive open online courses using hidden markov models. *Electrical Engineering and Computer Sciences University of California at Berkeley*, 2013.

[2] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287, 2002.

[3] Christopher G Brinton, Mung Chiang, Sonal Jain, HK Lam, Zhenming Liu, and Felix Ming Fai Wong. Learning about social learning in moocs: From statistical analysis to generative model. *Learning Technologies, IEEE Transactions on*, 7(4):346–359, 2014.

[4] Ed H Chi, Peter Pirolli, Kim Chen, and James Pitkow. Using information scent to model user information needs and actions and the web. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 490–497. ACM, 2001.

[5] Carleton Coffrin, Linda Corrin, Paula de Barba, and Gregor Kennedy. Visualizing patterns of student engagement and performance in moocs. In *Proceedings of the fourth international conference on learning analytics and knowledge*, pages 83–92. ACM, 2014.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):pp. 1–38, 1977.

[7] Andrew J Elliot and Todd M Thrash. Approach-avoidance motivation in personality: approach and avoidance temperaments and goals. *Journal of personality and social psychology*, 82(5):804, 2002.

[8] José P González-Brenes and Yun Huang. Using data from real and simulated learners to evaluate adaptive tutoring systems. In *Proceedings of the Workshops at the 18th International Conference on Artificial Intelligence in Education AIED*, 2015.

[9] James A Gordon, William M Wilkerson, David Williamson Shaffer, and Elizabeth G Armstrong. "practicing" medicine without risk: students' and educators' responses to high-fidelity patient simulation. *Academic Medicine*, 76(5):469–472, 2001.

[10] René F Kizilcec, Chris Piech, and Emily Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 170–179. ACM, 2013.

[11] Kenneth R Koedinger, Noboru Matsuda, Christopher J MacLellan, and Elizabeth A McLaughlin. Methods for evaluating simulated learners: Examples from simstudent. *17th International Conference on Artificial Intelligence in Education AIED*, 5:45–54, 2015.

[12] Nan Li, Łukasz Kidziński, Patrick Jermann, and Pierre Dillenbourg. How do in-video interactions reflect perceived video difficulty? In *Proceedings of the European MOOCs Stakeholder Summit 2015*, number EPFL-CONF-207968, pages 112–121. PAU Education, 2015.

[13] Ran Liu and Kenneth R Koedinger. Variations in learning rate: Student classification based on systematic residual error patterns across practice opportunities. In *Educational Data Mining 2015*. EDM, 2015.

[14] Gord McCalla and John Champaign. Aied 2013 simulated learners workshop. In *Artificial Intelligence in Education*, pages 954–955. Springer, 2013.

[15] Richard Nock and Frank Nielsen. On weighting clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(8):1223–1235, 2006.

[16] Mirko Raca, Łukasz Kidziński, and Pierre Dillenbourg. Translating head motion into attention-towards processing of student's body-language. In *Proceedings of the 8th International Conference on Educational Data Mining*, number EPFL-CONF-207803, 2015.

[17] Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daumé III, and Lise Getoor. Modeling learner engagement in moocs using probabilistic soft logic. In *NIPS Workshop on Data Driven Education*, 2013.

[18] Ipke Wachsmuth and Jens-Holger Lorenz. Sharpening one's diagnostic skill by simulating students' error behaviors. *Focus on learning problems in mathematics*, 9(2), 1987.

[19] Christopher A Wolters. Advancing achievement goal theory: Using goal structures and goal orientations to predict students' motivation, cognition, and achievement. *Journal of educational psychology*, 96(2):236, 2004.

Investigating Gender Differences on Homework in Middle School Mathematics

Mingyu Feng
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025
1-650-859-2756
mingyu.feng@sri.com

Jeremy Roschelle
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025
1-650-859-3049
jeremy.roschelle@sri.com

Craig Mason
University of Maine
5766 Shibles Hall
Orono, ME 04469
1-207-581-9059
craig.mason@maine.edu

Ruchi Bhanot
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025
1-650-859-5381
ruchi.bhanot@sri.com

ABSTRACT

Recent studies [10, 23] using US nationwide databases showed high school boys spent significantly less time doing homework than girls, based on their responses to questionnaires and surveys. To investigate gender differences in homework in middle school, in this paper, we analyzed computer log data and standardized test scores of more than 1,000 7th grade students who participated in a large-scale randomized controlled online homework efficacy study. Students used the ASSISTments platform to do their homework for a school year. Our results suggested no significant difference between the time the two genders spent on homework overall. There was a marginally significant difference on homework time between genders in the high performing group only. When examining the system-student interaction data, we found significant difference between boys and girls in their help-seeking behaviors. In addition, we found out that boys have benefited from the online homework intervention more than girls.

Keywords

Gender gap, homework, online homework intervention

1. INTRODUCTION

Studies have investigated gender differences in homework completion rates, learning habits, and technology use outside of school. The investigation into gender differences found that girls spend more time on homework [36], including math [28]. Further, research has also shown that girls are more likely to spend time regulating study habits (e.g., time management, engaging in emotion self-regulation while doing homework) [13, 16, 37, 38, 39]. This was especially true with girls receiving family help while doing homework [35, 36]. With regards to gender differences in technology out of school, research clearly indicates boys have an advantage over girls with using technology for more varied reasons (e.g., programming, gaming, or internet surfing) than girls (e.g., drawing) [33] and more frequently as well [12, 17, 22, 24, 27, 34]. This gender-based advantage extends to girls' attitudes towards computer usage. Girls tend to exhibit lower self-efficacy beliefs about their use of computers [21, 33]. At the same time, studies also document parent support as a critical mitigating factor that can increase girls' use of and experience with computers [21, 33].

More recently, two studies, [10] and [23] suggested that boys spend less time on homework than girls. Based on the PISA 2012 Database, [23] shows that around the globe, 15-year-old boys are overwhelmingly less likely than girls to spend time doing homework, which may in part explain why they are more likely to struggle academically. The study has been widely cited in recent press coverage (e.g. [26]). In the U.S., boys on average spend 1.8 hours less time per week doing homework than girls. When considering boys and girls who spend the same amount of time doing homework, the gender gap in mathematics achievement is wider. [10] examined data from American Time Use Survey (AUS) responses. They showed that high school girls spent statistically significantly more time (17 minutes per day) on homework than male high schoolers, even after controlling for SES indicators, daily activities and other factors. Furthermore, the gap for time spent on homework is largest among high-achieving students.

These studies illustrate that achievement gaps between genders' use of homework does exist. However, we noticed almost all studies on gender differences in homework use self-reported measures. PISA 2012 asked students to report how much time per week they spend doing homework by teachers. [10] used students' non-school study time using time diary data from 2003-2013 waves of the AUS and transcript data from the Educational Longitudinal Study of 2002 (ELS). Our literature search shows that there is a serious need for rigorous homework research on homework in K-12 settings. The existing studies are mostly correlational survey studies with thousands of students that relate homework time, academic-, and non-academic outcomes.

Our online homework study, which is a rigorously designed, randomized controlled experiment, gives us a unique opportunity to study the gender gap using more objective data sources of homework—computer logs from an online platform that support *middle school* students doing math homework. In this paper, examine the difference between genders in middle school mathematics on

- homework time
- the amount of problems completed by each gender
- homework performance
- how each gender interacted with the system
- whether there was any difference in the outcome measure between the two genders
- which gender benefited more from the technology-based intervention

2. BACKGROUND

2.1 Online homework study

Research has been conducted to study the role and practices of homework and its relationship with student learning, particularly for mathematics (e.g. [2, 3, 5, 8, 9, 19, 20, 28, 29]). The link between homework assignments and student achievement is far from clear across the board, as noted by Cooper and others [30]. Although some studies show that students—and especially struggling students—could benefit from middle school mathematics homework, they may not benefit under typical conditions. Technology-based learning environment, such as ASSISTments, provides ways to make homework more adaptive and productive for the students who could benefit most. These environments can also do some of the bookkeeping and help teachers to keep track the progress of their students. They enable teachers to assign customized homework to their students. For example, while doing homework in ASSISTments, students receive support including immediate feedback on the accuracy of their answers, as well as extensive tutoring. With these supports in place, students may complete more homework and learn more while doing homework. Teachers may be freed from the tedium of grading homework and be able to instead focus their energies adjusting and differentiating instruction.

SRI International, in conjunction with the University of Maine and Worcester Polytechnic Institute (the developer of the ASSISTments platform) conducted a multiyear randomized controlled efficacy trial at the school level. The study was conducted in 44 schools in the state of Maine, where one-to-one computing has been well-established for over 10 years. This experiment tested the hypothesis that the ASSISTments homework support improves student mathematics outcomes and will also examine impacts for struggling students and other important demographic groups. Schools in the study were randomly assigned to treatment or control (i.e. "business as usual") conditions. The intervention was implemented in 7th grade classrooms in treatment schools over 2 consecutive years. In the control condition, teachers and students continue with their existing homework practices. In the treatment condition, teachers received professional development and used ASSISTments in the first year to become proficient with the system and then teachers used ASSISTments with a new cohort of students in the second year which is considered the "experiment year". At the end of the experiment year, students were administered the TerraNova Common Core math test to provide data on student achievement in mathematics. TerraNova is a norm-referenced achievement test that is nationally normed. It generates scaled scores (ranging from 400 to 900 points) and achievement-level information that include five levels of performance proficiency (1: Starting-out; 2: Progressing; 3: Near Proficient; 4: Proficient; 5: Advanced).

2.2 Key features of ASSISTments platform

ASSISTments (www.assistments.org) [6, 14] is an online tutoring system that provides "formative assessments that assist." Teachers choose (or manually add) homework items in ASSISTments and students can complete their homework online. As students do homework in ASSISTments, they receive feedback on the accuracy of their answers. Some problem types provide hints to help students improve their answers, or help decompose multistep problems into parts (so-called "scaffolding questions")

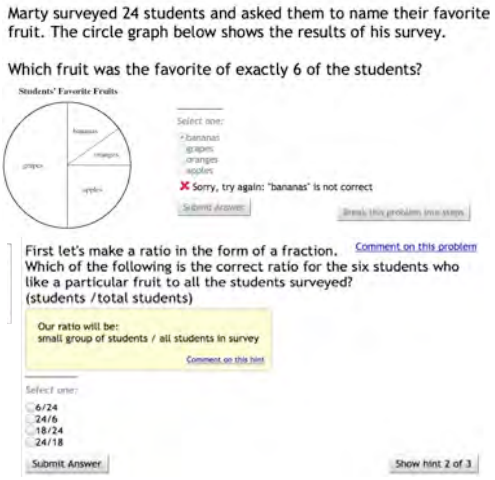


Figure 1. Screen shots of an 7th grade item in ASSISTments that provides correctness feedback and breaks the problem into steps.

(see Figure 1). Teachers may choose to assign problem sets called "skill builders" that address individual math concepts and skills at grade level and are organized to promote mastery learning. Every night, ASSISTments servers generate customized, cognitive diagnostic reports. The reports show teachers homework completion rates, performance data for each student on every problem and each math skill covered in the assignment, which questions and/or skills were particularly challenging for, and what the common wrong answers were. The report is emailed to teachers early in the morning for their review. This data allows teachers to make real-time, informed decisions about what and how they teach, and it is ideally used to guide homework review practices in class.

The usage model of the online homework study specifies that teachers who used ASSISTments in the study were expected to assign approximately 20 minutes of homework in ASSISTments for a minimum of three nights per week (making adjustments as needed to accommodate district and school homework policy).

3. EXPLORING GENDER DIFFERENCES IN HOMEWORK TIME, BEHAVIORS, AND PERFORMANCE

The data used in this section includes ASSISTments system logs of 1033 7th grade students, including 514 boys and 519 girls, who participated the second year of the homework study in the treatment condition. Also included in the data are their TerraNova test scores including both scaled scores and their performance levels. These students used ASSISTments for homework for the whole school year.

3.1 Features

We started the data analysis by constructing features that represent student's intensity of use, performance, and behaviors while working in ASSISTments. Below, we list all the features.

- mins_s: Total number of minutes students spent on homework in the year
- probs_c: Total number of problems completed
- perc: Average percent correct over all assignments
- hint_c: Average number of hint requests per problem

- attempt_c: Average number of attempts[1] per problem
- bottom_hint_c: Average number of bottom-out hint[2] requests per problem
- comp_perc: % of homework assignments completed on time
- late_perc: % of assignments completed but late

Two features, *mins_s* and *probs_c* are measures of intensity of use of ASSISTments. *perc* is a measure of student's performance on homework problems. Some other system features (*hint_c, attempt_c, and bottom_hint_c*) capture students' interaction with the system while doing homework, including their help-seeking behaviors (*hint_c* and *bottom_hint_c*) and the number of attempts they made before getting a correct answer (*attempt_c*). The last two features show whether they complete their homework on time or late (*comp_perc, late_perc*) as opposed to not completing an assignment at all.

3.2 Visual exploration of homework time

Research has shown that spending more time doing homework is better for academic achievement [3, 28, 30, 32]. Additional research has also shown that homework time is associated with many factors that may have a positive effect on academic success such as motivation or academic interest [4, 15] and parent involvement [1, 25]. Therefore, we started with an exploratory analysis focusing on the time students have spent on doing homework in ASSISTments. We observed relatively weak positive relationships[3] (.2 < r < .4) between students' TerraNova scaled scores and system use and performance indicators (*mins_s, probs_c,* and *perc*), suggesting students who spent more time on homework and completed problems scored higher on the TerraNova test. When we examined the usage data closely, we found that students spent a wide range of time on homework in ASSISTments in the school year (ranging from 2 to 4,238 minutes, mean = 640, standard deviation = 784), and amount of use varies a lot by schools (65% of the variance in *mins_s* is accounted by schools). Although homework practice is expected to differ across teachers and schools, the large variance is to some extent surprising, as the research team has specified a desired use model and has expressed the expectations clearly to all teachers in the treatment schools. On the other hand, this result confirms our previous findings on implementation fidelity from the previous 2013-14 school year where adherence, exposure, and uptake of users varied by teachers [7].

Next, we further explored the relationship between homework time and students' achievement outcomes. We found that higher-performing students tend to spend more time on homework. Girls seem to spend relatively more time on homework than boys do, except in the middle level of achievement. The difference is most notable in the "5: Advanced" level.

Then we compared the TerraNova performances of boys and girls who spend similar amounts of time on homework. We found that there are more girls than boys who spent a significant amount of time on homework (defined as over 3,200 minutes in the school year). Unlike [23], however, we didn't see big gender gaps in mathematics achievement (Figure 3).

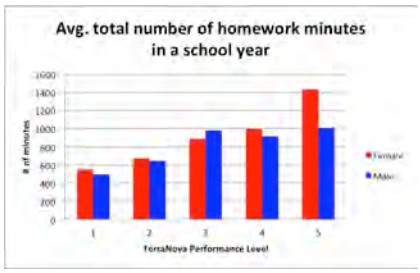


Figure 2. Bar graph comparing the average homework time by students in each TerraNova performance level, split by gender

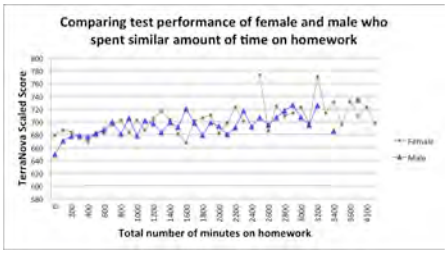


Figure 3. Plot comparing TerraNova performance of boys and girls who spend similar amount of time on homework

3.3 Modeling gender difference on usage and homework performance

Table 1 shows the descriptive statistics of all the features by gender. We noticed that the mean difference between the two genders were high on the two features, *mins_s* and *probs_c*, yet standard deviations on those measures were also quite high. We understood the extent to which schools create variation in homework behaviors: differences in the amount of homework assigned between teachers and schools, possible variations in homework review processes, and differences in teachers' completion policies. Since these factors could affect students' performance and/or behavior when doing homework, we trained a series of 3-Level Hierarchical Linear Regression models (HLM) (students nested in classes and classes in schools) to account for the difference in schools' and teachers' homework assignment practices. We used each feature as a dependent variable and use *gender* of students as the predictor (male = 0, female = 1).

Table 1. Descriptive statistics of features by gender

Features	Male		Female	
	Mean	Stdev	Mean	Stdev
mins_s	820.337	759.742	874.755	807.623
probs_c	703.214	592.099	770.734	621.226
perc	0.740	0.115	0.744	0.117
hint_c	0.115	0.143	0.094	0.103
attempt_c	1.403	0.281	1.375	0.248
bottom_hint_c	0.072	0.074	0.061	0.068
comp_perc	0.614	0.284	0.646	0.259
late_perc	0.129	0.12	0.14	0.127

As shown in Table 2, the results suggest that overall there is no significant difference between girls and boys in terms of the amount of time they spent on homework or the number of problems they completed. Furthermore, there is no difference between the two genders in their rates of correctly answered

[1] The system doesn't limit the number of answers a student could attempt on a problem.

[2] When using ASSISTments in the practice and learning modes (as opposed to testing mode), the system requires that every problem has to be answered correctly in order for students to move to the next one. Bottom-out hints in ASSISTments reveal the correct answer to students so that they won't get stuck.

[3] No other correlations were noticed

problems in ASSISTments. Girls tend to complete more assignments on time than boys, but the difference is only marginally significant (p = .086). However, interestingly, girls and boys interacted with the system differently; girls made fewer hint requests and fewer attempts on problems, and they also requested fewer bottom-out hints as compared to boys in the same classes.

Table 2. HLM Results Overall – Predictor: Female

Dependent Variable	Difference	*p*
mins_s	22.482	0.350
probs_c	27.219	0.177
perc	0.006	0.351
hint_c	-0.018	0.005**
attempt_c	-0.039	0.005**
bottom_hint_c	-0.011	0.002**
comp_perc	0.015	0.086-
late_perc	-0.000	0.907

Inspired by Gershenson & Holt (2015) and Figure 3 shown above, we were interested to see whether there was any interaction effect between gender and students' performance levels. Thus, we split the students into 3 groups based on their performance level on the TerraNova test. We then trained similar HLM models within each group of students, and the results are shown in Table 3.

- **Progressing or Below**: performance levels = 1 or 2; N= 328 (male: 145, female: 183)
- **Near Proficient**: performance levels = 3; N = 368 (male: 165, female: 203)
- **Proficient or Above**: performance levels = 4 or 5); N = 337 (male: 166, female: 171)

We observed trends with regard to how students interact with the system in both the *Near Proficient* and *Proficient or Above* groups. The trends are consistent with the overall trend: girls requested significantly fewer regular hints or bottom-out hints, and made fewer attempts on problems. Results regarding assignment completion status are mixed. Low-performing girls completed fewer assignments after they were due than low-performing boys did; yet in the *Near Proficient* group, girls completed more assignments late than boys did. In the *Proficient or Above* group, girls were more likely to complete assignments on time. Interestingly, we noticed a marginally significant difference in *mins_s* in the *Proficient or Above* group, suggesting high-performing girls spent more time on homework than high performing boys. This result is in consistent with [10], but the latitude of difference is not as big.

4. WHICH GENDER BENEFITED MORE FROM TECHNOLOGY-BASED HOMEWORK INTERVENTION?

One of the research questions of the online homework study is to investigate whether the impact of the ASSISTments on learning outcomes differ by student demographic characteristics. Here we present the analysis that was conducted to examine which gender benefited more from online homework intervention. A different dataset was used for this analysis. Students from the control condition were included in this dataset in order to detect the interaction between intervention and gender, which increased the total number of students to 2,756 from 44 schools. Only students' assigned condition, gender, their 6th grade state test scores, and their TerraNova scaled scores were included in this dataset. TerraNova scores were used as dependent variable. 3-level HLM models were employed in all the analysis.

We first ran a basic model that includes prior achievement (6th grade math state test scores) and gender (male=1, female=0) as predictors of TerraNova scaled scores to examine effects of gender. The HLM model for the analysis of effect of gender is illustrated below.

Level-1 model:

$$TScore = \beta_{0j} + \beta_{1j}*(PriorMath) + \beta_{2j}*(Male) + r$$

Level-2 model:

$$\beta_{0j} = \gamma_{00} + \gamma_{01}*(Trx) + u_0$$
$$\beta_{1j} = \gamma_{10}$$
$$\beta_{2j} = \gamma_{20}$$

In this model, TScore is the student's scaled score from the TerraNova test. *Trx* is a school-level indicator of the school being in the treatment condition (0=Control, 1=Treatment). Student-level variables. *PriorMath* is a student-level variable, representing the student's 6th grade math state test score. *Male* is a student-level variable, indicating the student's gender (0=Female, 1=Male). The model showed that students in the treatment condition scored 10.26 points higher than control students and males scored 5.21 points lower than females. Both effects are statistically significant ($p < .001$). To help understand the difference, we referred to TerraNova technical norms published by CTB. The norms showed that the average yearly growth from 7th to 8th grade is about 10 points in scaled score.

Table 4. HLM Results on Intervention and Gender Effect

Gender	Control	Treatment	Difference
Females	683.21	693.46	10.26
Males	677.99	688.25	10.26
Difference	-5.21	-5.21	

Then we augmented the basic model by adding an interaction term between treatment and gender. The augmented model is illustrated below.

Level-1 model:

$$TScore = \beta_{0j} + \beta_{1j}*(PriorMath) + \beta_{2j}*(Male) + r$$

Level-2 model:

$$\beta_{0j} = \gamma_{00} + \gamma_{01}*(Trx) + u_0$$
$$\beta_{1j} = \gamma_{10}$$
$$\beta_{2j} = \gamma_{20} + \gamma_{21}*(Trx) + u_2$$

Table 3. HLM Results By Groups – Predictor: Female

Dependent Variable	Progressing or Below		Near Proficient		Proficient or Above	
	Difference	*p*	Difference	*P*	Difference	*p*
mins_s	8.565	0.859	-18.195	0.684	58.401	0.073-
probs_c	34.421	0.378	-17.773	0.638	34.694	0.178
perc	-0.008	0.569	0.005	0.632	0.013	0.058-
hint_c	-0.015	0.239	-0.024	0.061-	-0.012	0.064-
attempt_c	-0.017	0.56	-0.066	0.005**	-0.033	0.075-
bottom_hint_c	-0.003	0.685	-0.016	0.007**	-0.013	0.002**
comp_perc	0.021	0.185	-0.001	0.929	0.027	0.055-
late_perc	-0.021	0.032*	0.018	0.034*	-0.005	0.521

In this model, the effect of ASSISTments intervention was found to vary by gender (γ_{21}=7.476, t(42)=2.232, p = 0.031). As shown in Table 5, boys in the control group scored 9.61 points lower than girls in the control group, but boys in the treatment condition scored only 2.13 points lower than girls in the same group. Girls in the treatment group scored 6.73 points higher than those in the control group (which was not significant after adding in the interaction), while boys in the treatment group scored 14.21 points higher than those in the control group. In essence, the intervention helped close the gender gap between girls and boys for standardized test achievement and boys have benefited more from the intervention than girls.

Table 5. HLM Results on Intervention and Gender Interaction Effect

Gender	Control	Treatment	Difference
Females	685.20	691.93	6.73
Males	675.59	689.79	14.21
Difference	-9.61	-2.13	

5. CONCLUSIONS AND FUTURE STUDIES

In this paper, we examined the difference between genders in middle school mathematics on homework time, the amount of problems completed by each gender, homework performance, and how each gender interacted with the system, using computer system log data from an online homework intervention. We also answered two research questions regarding which gender benefited more from a technology-based intervention supporting homework. Our results suggested no significant difference between the time the two genders spent homework overall. Among students who performed proficiently or above on the end-of-year standardized test, girls have spent more time on homework than boys, and the difference was marginally significant. We also found out that when using ASSISTments for homework, girls and boys differed in their help-seeking and problem-attempting behaviors. Girls requested less hints, made less number of attempts on problems, and they also requested less amount of bottom-out hints that would reveal the correct answers to problems. Our findings suggested that the intervention closed gender gaps in mathematics achievement in 7th grade and boys benefited from the online homework intervention more than girls.

We speculated on the reasons why boys have benefited more from the technology-based intervention. One reason could be boys in the study were more comfortable with using technologies, similar to what has been reported in earlier research. We also checked to see if there was any difference between the two genders in prior achievement. Using a simple *t*-test, there was no gender difference in 6th grade state math test scores (Female average score =645, Male average score=644, p =0.252).

Researchers have been able to identify factors that impact this relationship between time spent doing homework and academic achievement. It was found that the quality of time spent on a task, i.e., homework, is a more critical predictor of student learning than the total number of minutes spent on the task. For instance, time on task or perseverance manifested with low distraction rates is positively correlated with achievement [30]. Other factors, especially the effort students put into homework and how frequently they do homework are far more reliable and positive predictors of student achievement [28, 30, 32]. As a follow-up study, we plan to look at student behaviors when working in in the system more closely, taking sequence and time into account. We plan to study help-seeking and problem-attempting behaviors at action level and to see whether there are any the sequential pattern of actions taken, and whether there is between girls and boys. For instance, did boys ask for hints/bottom-out hints right away, while girls took time to persevere through challenging homework problems before requesting for assistance? We also plan to build a dataset including students' frequency of logging in each day and each week and the duration of the working sessions by gender, and see how such features predict student learning. Such studies will help the field better understand gender differences in STEM learning, esp. in out-of-classroom settings. The findings can be informative for the development of behavior detectors in online learning systems like ASSISTments so that the systems can provide interventions to improve learning outcomes and close gender gaps.

We recognize the limitations in our study. We have no access to information, such as parent involvement, their extra-curricular activities, etc. that may affect student's homework completion rates, their behaviors when doing homework, or their performance. Nor do we have access to student's attitudes towards mathematics, technology or homework. All of these limit our ability to explain the differences we've discovered. The results presented in this paper were based on data from 7th grade students who are younger than the high school students who have been the focus of [23] and [10]. It would be a reasonable next step to extend such kind of study to elementary students and see if there might exist a trajectory in the gender differences in homework.

6. ACKNOWLEDGMENTS

This material is based upon work supported by the Institute of Educational Sciences (IES) of U.S. Department of Education under Grant Number R305A120125. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the IES.

7. REFERENCES

[1] Bhanot, R., Jovanovic, J. (2005). Do parents' academic gender stereotypes influence whether they intrude on their children's homework? Sex Roles, 52(3)(9/10), 597-607.

[2] Cooper, H. (2007). The battle over homework (3rd ed.). Thousand Oaks, CA: Corwin Press.

[3] Cooper, H., Robinson, J. C., & Patall, E. A. (2006). Does homework improve academic achievement? A synthesis of research, 1987–2003. *Review of Educational Research, 76*(1), 1–62.

[4] Eccles, J. & Wigfield, A. (2002). Motivational beliefs, values, and goals. *Annual Review of Psychology, 53*(1), 109-132.

[5] Eren, O., & Henderson, D. (2011). Are we Wasting Our Children's Time by Giving Them More Homework? *Economics of Education Review*, 30(5), 950-961.

[6] Feng, M., Heffernan, N., and Koedinger, K. (2009). Addressing the assessment challenge in an Online System that tutors as it assesses. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research (UMUAI journal)*. 19(3), 243-266, August, 2009.

[7] Feng, M., Roschelle, R., Murphy, R. & Heffernan, N. (2014). Using Analytics for Improving Implementation Fidelity in a Large Scale Efficacy Trial. In *Proc. ICLS 2014.* International Society of the Learning Sciences. pp. 527-534.

[8] Fernández-Alonso, R., Suárez-Álvarez, J., & Muñiz, J. (2015, March 16). Adolescents' Homework Performance in Mathematics and Science: Personal Factors and Teaching Practices. *Journal of Educational Psychology*.

[9] Galloway, M. K., & Pope, D. (2007). Hazardous homework? The relationship between homework, goal orientation, and well-being in adolescence. *Encounter*, 20, 25–31.
[10] Gershenson, S. & Holt. S. (2015). Gender gaps in high school students homework time. *Education Researcher*, Voc. 44. No. 8. Pp432-441.
[11] Gill, B. & Schlossman S. (2003). A Nation At Rest: The American Way of Homework. *Educational Evaluation and Policy Analysis*, 25(3).
[12] Hakkarainen, K., Ilo¨maki, L., Lipponen, L., Muukkonen, H., Rahikainen, M., Tuominen, T., et al. (2000). Students' skills and practices of using ICT: Results of a national assessment in Finland. Computers and Education, 34(2), 103–117.
[13] Harris, S., Nixon, J., & Rudduck, J. (1993). School work, homework and gender. *Gender and Education, 5*(1), 3-14.
[14] Heffernan, N. & Heffernan, C. (2014). The ASSISTments Ecosystem: Building a Platform that Brings Scientists and Teachers Together for Minimally Invasive Research on Human Learning and Teaching. International Journal of Artificial Intelligence in Education. 24(4), 470-497.
[15] Hidi, S., & Renning, K.A. (2006). The four-phase model of interest development. *Educational Psychologist, 41*(2), 111-127.
[16] Honigsfeld, A., & Dunn, R. (2003). High school male and female learning-style similarities and differences in diverse nations. *Journal of Educational Research, 96*(4), 195-206.
[17] Janssen Reinen, I. J., & Plomp, T. (1997). Information technology and gender equality: A contradiction in terminis? Computers and Education, 28(2), 65–78.
[18] Juster, T. F., Ono, H., & Stafford, F. P. (2004). *Changing Times Of American Youth: 1981-2003.* Institute for Social Research University of Michigan.
[19] Maltese, A.V., Robert, H.T., and Fan, X. (2012). When Is Homework Worth the Time? Evaluating the Association Between Homework and Achievement in High School Science and Math. *The High School Journal*, October/November 2012: 52-72.
[20] Marzano, R. J., & Pickering, D. J. (2007). The case for and against homework. Educational Leadership, 64, 74–79.
[21] Meelissen, M. R. M., & Drent, M. (2007). Gender differences in computer attitudes: Does the school matter? Computers in Human Behavior. doi:10.1016/j.chb.2007.03.001.
[22] Nelson, L. J., & Cooper, J. (1997). Gender differences in children's reactions to success and failure with computers. Computers in Human Behavior, 13(2), 247–267.
[23] OECD (2015). *The ABC of Gender Equality in Education: Aptitude, Behavior, Confidence*. PISA, OECD Publishing. http://dx.doi.org/10.1787/9789264229945-en
[24] Papastergiou, M., & Solomonidou, C. (2005). Gender issues in internet access and favourite internet activities among Greek high school pupils inside and outside school. Computers and Education, 44(4), 377–393.
[25] Ramey, G & Ramey, V. (2010). The rug rat race. *Brookings Papers on Economic Activity, 41*(1), 129-199.
[26] Rushhowy, K. (March, 2015) Retrieved from http://www.ourwindsor.ca/news-story/54609 64-boys-do-less-homework-than-girls-global-study-finds/
[27] Selwyn, N. (1998). The effect of using a home computer on students' educational use of IT. Computers and Education, 31(2), 211–277.
[28] Trautwein, U. (2007). The homework-achievement relation reconsidered: Differentiating homework time, homework frequency, and homework effort. Learning and Instruction, 17, 372–388. doi: 10.1016/j.learninstruc.2007.02.009.
[29] Trautwein, U., Koller, O., Schmitz, B., & Baumert, J. (2002). Do homework assignments enhance achievement? A multilevel analysis in 7th-grade mathematics. *Contemporary Educational Psychology*, 27.1: 26-50.
[30] Trautwein, U., & Koller, O. (2003a). The relationship between homework and achievement: still much of a mystery. Educational Psychology Review, 15, 115e145.
[31] Trautwein, U., & Koller, O. (2003b). Time investment does not always pay off: the role of self-regulatory strategies in homework execution. Psychologie, 17, 199e209.
[32] Trautwein, U., Ludtke, O., Schnyder, I., & Niggli, A. (2006). Predicting homework effort: support for a domain-specific, multilevel homework model. Journal of Educational Psychology, 98, 438e456.
[33] Vekiri, I., & Chronaki, A. (2008). Gender issues in technology use: Perceived social support, computer self-efficacy and value beliefs, and computer use beyond school. *Computers & education, 51*(3), 1392-1404.
[34] Volman, M., van Eck, E., Heemskerk, I., & Kuiper, E. (2005). New technologies, new differences. Gender and ethnic differences in pupils' use of ICT in primary and secondary education. Computers and Education, 24(1), 35–55.
[35] Xu, J. (2006). Gender and Homework Management Reported by High School Students. *Educational Psychology, 26*(1), 73-91. doi: 10.1080/01443410500341023
[36] Xu, J. (2007). Middle-School Homework Management: More than just gender and family involvement. *Educational Psychology, 27*(2), 173-189.
[37] Xu, J., & Corno, L. (2006, March 10). Gender, family help, and homework management reported by rural middle school students. *Journal of Research in Rural Education, 21*(2). Retrieved [date] from http://jrre.psu.edu/articles/21-2.pdf.
[38] Younger, M., & Warrington, M. (1996). Differential achievement of girls and boys at GCSE: Some observations from the perspective of one school. *British Journal of Sociology of Education, 17*(3), 299-313.
[39] Zimmerman, B. J., & Martinez-Pons, M. (1990). Student differences in self-regulated learning: Relating grade, sex, and giftedness to self-efficacy and strategy use. *Journal of Educational Psychology, 82*(1), 51-59.

Investigating Difficult Topics in a Data Structures Course Using Item Response Theory and Logged Data Analysis*

Eric Fouh
Department of Computer Science & Engineering
Lehigh University
Bethlehem, PA 18015
efouh@cse.lehigh.edu

Mohammed F. Farghally
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
mfseddik@vt.edu

Sally Hamouda
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
sallyh84@vt.edu

Kyu Han Koh
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
kyuhan@vt.edu

Clifford A. Shaffer
Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
shaffer@vt.edu

ABSTRACT

We present an analysis of log data from a semester's use of the OpenDSA eTextbook system with the goal of determining the most difficult course topics in a data structures course. While experienced instructors can identify which topics students most struggle with, this often comes only after much time and effort, and does not provide real-time analysis that might benefit an intelligent tutoring system. Our factors included the fraction of wrong answers given by student, results from Item Response Theory, and the rate of model answer and hint use by students. We grouped exercises by topic covered to yield a list of topics associated with the harder exercises. We found that a majority of these exercises were related to algorithm analysis topics. We compared our results to responses given by a sample of experienced instructors, and found that the automated results match the expert opinions reasonably well. We investigated reasons that might explain the over-representation of algorithm analysis among the difficult topics, and hypothesize that visualizations might help to better present this material.

Keywords

Item Response Theory, learning analytics, eTextbooks, algorithm analysis, data structures and algorithms

*(Does NOT produce the permission block, copyright information nor page numbering). For use with ACM_PROC_ARTICLE-SP.CLS. Supported by ACM.

1. INTRODUCTION

Knowing what topics are challenging to students helps educators better allocate course resources. We present techniques to automatically determine topics that are most challenging based on student interactions within the OpenDSA eTextbook system [9, 10]. While experienced instructors can identify which topics students most struggle with, automated measures can be useful for a variety of reasons. 1) Identifying key topics takes a lot of time and effort on the part of instructors; 2) They can help instructors teaching new material or with a new approach; 3) They can be used by an intelligent tutoring system (ITS) to automatically direct more instruction to a topic; and 4) They can help find, confirm, and quantify relationships and provide new insights that might be missed even by experienced instructors.

Our study focuses on a post-CS2 data structures and algorithms course (henceforth referred to as "CS3"). We used two approaches to identify difficult course topics. The first is Item Response Theory (IRT), a latent trait models (LTM) technique to analyze student responses to problems. LTM assumes that test performance can be predicted by specific traits or characteristics [13]. IRT provides a model-based association between item responses and the characteristic assessed by a test [7]. The second approach consisted of an analysis of student interactions with exercises to identify harder exercises. We investigated the incidence of guessing, the use of hints, and the level of interactions with embedded model answers by students when solving exercises.

We found that the most difficult topics in the CS3 course are related to algorithm analysis. While this is not surprising to us, we also investigated possible reasons that might explain the topics' difficulty. Based on our study, we present some suggestions on how to make such topics more accessible to students.

2. RELATED WORK

IRT [19] examines test behavior at the item level, and provides feedback on the relative difficulty of the various ques-

tions. Many IRT models have been developed with the assumption of 0 or 1 assigned to each response. We adopted the one parameter (1PL) or Rasch model [16] to characterize items and examinees. In 1PL, the probability of a positive response from a student is a function of item difficulty and is modeled as $P_i(\theta) = \frac{\exp(\theta - b_i)}{1+\exp(\theta - b_i)}$. P_i is the probability of a correct response to item i. θ refers to the latent trait (this is often called *ability*) assessed by the items being analyzed. b_i represents the difficulty of item i.

IRT has been used to evaluate students' coding abilities in an introductory programming course [3]. The authors used students' code scores to build a 1PL Rasch model. They found that students with previous knowledge had a statistically significant higher performance than students with no previous knowledge [3]. IRT was also used to analyze midterm exam questions for an introductory CS course [18]. The goal was to improve the assessment for future semesters by studying questions' item characteristic curves. IRT has been used for problem selection and recommendation in ITS [14]. The authors built a model based on a combination of IRT and collaborative filtering to automatically select problems.

We know of few efforts to identify difficult topics in CS3 courses, as most such work typically has focused on introductory courses [5, 6, 11]. Brusilovsky et al [4] sent a questionnaire to CS educators asking them to report topics that they consider critical to learn, as well as topics that are hard to learn (for students) and hard to teach (for instructors). Instructors' ($n = 61$) five most difficult-to-learn topics included pointers, recursion, polymorphism, memory allocation, and parameter passing. The five most difficult to teach topics included recursion, pointers, error handling, algorithms, and polymorphism. Many of these topics are covered in CS3, but it is typically not the first time that students will have seen them.

3. EXERCISE ANALYSIS

OpenDSA provides a collection of online, open-source tutorials that combine textbook-quality text with algorithm visualizations, randomly generated instances of interactive examples, and exercises to provide students with unlimited practice. Content within OpenDSA is organized into modules, each focusing on a specific topic such as Quicksort or Closed Hashing. The modules contain a wide variety of exercises. Some require that the student manipulate a data structure to show the changes that an algorithm would make on it. We refer to these as "proficiency exercises" (PE exercises). This type of exercise was pioneered in the TRAKLA2 system [15]. OpenDSA uses the Khan Academy (KA) exercise framework [1] to provide multiple choice, T/F, and short answer exercises. We also use the KA framework to implement simpler proficiency exercises.

We studied 143 student participants enrolled in a CS3 course at Virginia Tech during Fall 2014. OpenDSA was used as the main textbook, and students had until the end of the semester to complete the OpenDSA exercises. OpenDSA exercises accounted for 20% of the course final grade.

[1] `http://github.com/Khan/khan-exercises`

3.1 Analysis of correct answer ratios

Our goal is to assign a value to each OpenDSA exercise in terms of "relative difficulty". We seek to find which exercises are relatively difficult for average ability students. From this, we hope to deduce which topics are most difficult for students. This in turn might lead us to refocus our instructional efforts, or come up with new interventions and presentation approaches. Unfortunately, it is not a simple matter to tell whether a question is difficult. OpenDSA works on a mastery-based system, meaning that students can repeat a question until they get it correct. As a result, most students earn full credit on almost all exercises. To confuse the situation further, as is typical with online courseware, some exercises can be "gamed" [1]. In our case, this happens when students repeatedly reload the current page until they get an easier problem instance to solve (though the system is implemented in ways to discourage other forms of guessing on any given question [9]). For these reasons, we cannot simply count how many students got an exercise correct. Instead, we developed alternative definitions for difficulty.

We analyzed OpenDSA exercises with respect to the ratio of correct to incorrect answers as a measure of exercise difficulty, that is, harder exercises should show a lower correct attempt ratio. To assess student performance, we use the fraction $r = \frac{\#\text{of correct attempts}}{\#\text{of total attempts}}$. For each exercise, we compute the difficulty level (dl) as $dl = 1 - \frac{\sum_{i=1}^{n} r_i}{n}$ where n is the number of students and r is the ratio of correct attempts. Similar metrics have been used previously to assess exercise difficulty. In [2], the authors used "how many attempts it takes for a student to determine the correct answer once they have made their initial mistake" as a measure of exercise difficulty for logic exercises. History of attempts coupled with IRT was also used in [17] to estimate exercise difficulty for an ITS.

We ranked the exercises by their dl and grouped them into quartiles. Dl scores ranged from 0 to 0.72. Exercises in the 4^{th} quartile ($dl > 0.25$) consist mainly of exercises covering concepts related to algorithm analysis (22 out of 26 in that quartile), and one was a code writing question. Exercises in the 3^{th} quartile ($0.13 \leq dl \leq 0.25$) covered mainly (14 out of 25) the mechanics of algorithms or data structures. Ten of these exercises covered course concepts. Exercises in the 2^{nd} quartile ($0.05 \leq dl < 0.13$) covered mainly (23 out of 25) the mechanics of algorithms or data structures. The other two were summary exercises covering lists and the introduction chapter. All exercises in the 1^{st} quartile ($dl < 0.05$) covered algorithms or data structures mechanics. These results indicate that students did not seem to have difficulty completing tasks related to the behavior and the mechanics of algorithms and data structures. They seem to have the hardest time mastering algorithms analysis concepts.

3.1.1 IRT analysis

To perform IRT analysis we must dichotomize the answers. We awarded 1 point for $r \geq 0.75$ and 0 point for $r < 0.75$. We analyzed each chapter independently, considering all exercises in a chapter as part of an assignment. We used R statistical software (ltm package) and built a 1PL model for our investigation. For each OpenDSA chapter, we computed the item characteristic curves (ICC), item information curves

(IIC), and test information curves (TIF). For each curve, the $x-axis$ represents the students' ability from -4 to 4, where $x = 0$ means average ability. ICC shows the probability of a score of 1, given a student's ability. IIC shows how much information each exercise can tell us about a students' ability. TIF shows how reliable the overall test (or a collection of exercises) is at distinguishing students with different ability. Harder tests would better distinguish between students with above-average ability, while easier tests would better distinguish between students with below-average ability.

An ICC graph lets us see the probability of getting a score of 1 for students with average ability. Harder exercises will have $P_i(0) < 0.5$. In Figure 1, we see that for three of the most difficult exercises, the probability that a student with average ability will get a score of 1 is less than 0.5, indicating that those exercises distinguish students with average ability from those with above average ability, but do little to distinguish weaker from average students. On the other hand, an easy question on the binary search algorithm has a graph $P_i(\theta) = 1$. Thus it does not give us any information about students' ability. The curves for the easier exercises shown in Figure 2 show differences between students with below average ability in contrast with average and above average ability ($\theta \geq 0$). Another possible interpretation of this result is that these exercises are relatively good at differentiating students who studied from those who did not. The TIF graph is a combination of all IIC curves, and indicates the overall performance of the test.

Algorithm analysis chapter exercises: Most students did not fare well on exercises in the introductory chapter on algorithm analysis, as shown in Figures 1 and 2. Thus these exercises gave us information about which students have above-average ability.

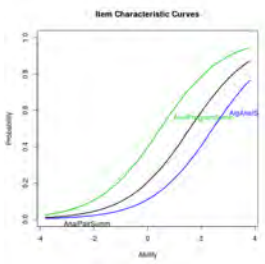


Figure 1: Algorithm analysis ICC

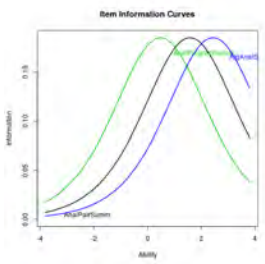


Figure 2: Algorithm analysis IIC

Linear Structures exercises: These students were already familiar with linear structures, since these are taught in prerequisite courses. Students could easily get a score of 1 by our difficulty measure for most problems in this chapter, and so help to identify students with below average ability ($x < 0$). However, three exercises appeared to be not so easy for students. They covered list overhead concepts (a new topic for them), array list concepts, and a small programming exercise. Students who did poorly (bottom quarter) on these exercises scored an average 65 on Midterm 1 compared to 76 for the rest of the class (a significant difference at $\alpha = 0.05$). They received an average score of 73 on Midterm 2 compared to 79 for the rest of the students (a significant difference at $\alpha = 0.05$). They scored an average

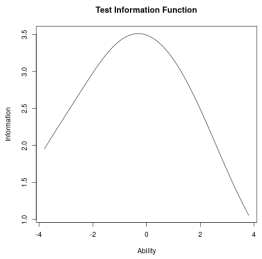


Figure 3: Sorting TIF

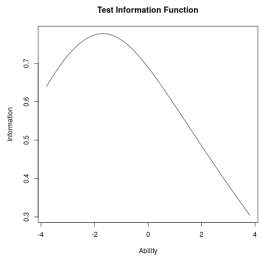


Figure 4: Binary trees TIF

of 106 on the final, compared to 112 for the rest of the class, not a statistically significant difference.

Sorting exercises: The sorting chapter has the most exercises, with varying difficulty levels. Summary exercises covering more advanced sorting algorithms (quicksort, radix sort, mergesort, and heapsort) seemed to provide more information about students with above average ability ($x > 0$). Overall, the sorting chapter exercises seemed to provide a good range of easy to difficult exercises, and provided good information to distinguish between students with different ability levels (TIF curve maximum at ability $= 0$).

Binary tree exercises: Binary trees are typically first introduced in CS2 courses. Only three exercises appeared to be difficult for students. These involved writing a recursive function to traverse a tree, questions on heaps, and computing tree space overhead. Exercises in this chapter provided us with information about students with below average ability (TIF curve maximum at ability < 0).

Hashing and graph exercises: As with other topics, proficiency exercises were relatively easy for the students, while questions on the concepts and analysis were more difficult. The graph chapter only had algorithm proficiency exercises and so were not challenging to students. Therefore, the exercises gave us information to distinguish students with low ability (TIF curve maximum at ability > 0).

We identified 21 (out of 100) exercises with IIC maximum at ability ≥ 0. 19 of those exercises cover the algorithm analysis portions of the different topics. The IRT analysis for all OpenDSA exercises given to students enrolled in the course revealed the following. Across chapters, exercises related to algorithm analysis had IIC curve maximums at ability< 0. Exercises that required students to solve small programming problems also scored as "difficult" by our metric because they tended to require multiple submissions to complete.

3.2 Using Hints and Guessing

Our analysis metric for "incorrect attempts" does not differentiate between using a hint or submitting an incorrect answer. So we looked in more detail at the types of incorrect submission for each exercise. We analyzed OpenDSA exercises with respect the the number of hints used, and the appearance of a trial-and-error strategy to "guess" the answers. Harder exercises are expected to display a higher rate of hints use and/or trial-and-error.

Exercises using the KA framework (multiple choice, T/F, fill-in-the-blank, and one-step proficiency exercises) generate a series of question instances on the topic. The student must get a certain number correct (typically five) to complete the exercise. One point is deducted from the student's credit toward this requirement when they submit an incorrect answer, to discourage guessing. Students can also take one or more hints that explain the answer to the question. In this case, the attempt is not graded (no point is awarded or deducted toward the threshold).

To analyze exercises based on students' hint use, we computed the hint ratio $hr = \frac{\#\text{of hints used}}{\#\text{of total attempts}}$ for each KA exercise. Four exercises are potential outliers as measured by hr, related to quicksort, hashing, calculating overhead for trees, and calculating overhead for lists. To analyze exercises based on the rate of trial-and-error, we calculated the incorrect ratio $ir = \frac{\#\text{of incorrect answers}}{\#\text{of total attempts}}$ for each KA exercise. Inspecting exercises in the fourth quartile (exercises in the highest 25% incorrect ratio), we found that they are related to the topics algorithm analysis, heaps, quicksort, radixsort, shellsort, and heapsort.

The seven exercises shown in Table 1 had high hint or high incorrect answer ratios. They relate to topics covering mathematical background and runtime analysis of quicksort, hashing, and shellsort. 45% of students heavily (third quartile and up for all exercises) used hints, and provided many incorrect answers when solving these seven exercises. We found that most exercises with low incorrect answer and hint ratios are for stacks, arrays, and lists. These are topics that most students know from previous courses. When using high rate of hint use as a measure of exercise difficulty, we found that exercises related to algorithm analysis and mathematics topics appeared to be more "difficult". Algorithm analysis was also identified as difficult by IRT analysis.

Table 1: *IR* and *HR* for difficult exercises

Exercise	*hr*	*ir*	Topic
ListOverhead	0.93	0.6	List Overhead Analysis
TreeOverheadSumm	0.78	0.73	Tree Overhead Analysis
QuicksortSumm	0.32	0.67	Quicksort Analysis
AlgAnalSumm	0.24	0.77	Algorithm Analysis
MthBgSumm	0.25	0.63	Mathematical background
ShellsortSumm	0.16	0.61	Shellsort
QuicksortPartitionPRO	0.27	0.58	Quicksort's partition

3.3 Model Answer Use and Exercise Reset

Algorithm proficiency exercises require students to reproduce the major steps of an algorithm. Proficiency exercises come with a "model" answer that can be viewed at any time (though doing so voids that problem instance for credit, and so the student must do another problem instance). The student can click a "reset" button to get a new problem instance. We analyzed OpenDSA exercises with respect to model answer use and "reset" as a measure of (exercise) difficulty. Students are expected to reset or view model answers more for harder exercises. For each proficiency exercise, we analyzed the number of student attempts and the frequency of student access to the model answer dialog. Our analysis showed that heap and quicksort exercises have a model answer view rate approaching or exceeding 50%, which is greater than the mean ($\mu = 25.5$) plus one standard deviation ($\sigma = 16$) of the rates distribution. This finding indicates that these exercises are relatively more difficult compared to other proficiency exercises.

We also investigated student activity log data to learn when students access the model answer box by computing: (i)% of students who tried the exercise, then opened the model answer dialog before they received enough points to get credit for the exercise; % of students who opened the model answer dialog before attempting the exercise; and % of students who opened the model answer dialog after they received proficiency credit for the exercise.

A model answer shows how to solve a problem with less detail, while slideshows and visualizations (available to the students before attempting the exercise) carefully explain the concepts. We tried to determine if students use model answers as a substitute for viewing slideshows and visualizations. For the heap exercises, we found that about 35% of the students attempted an exercise before going through any slideshow included in the section. This result indicates that students might be using model answers (on certain topics) because they overlook and/or rush through visualizations when studying. We found that a majority of students (62% on average) opened the model answer before attempting the heap exercises. For the quicksort exercise, we found that most students (67%) opened the model answer dialog after an incorrect attempt. 24% of students opened the model answer dialog before attempting the exercise.

For each proficiency exercise, we looked at the percentage of students who returned back to solve the exercise after receiving proficiency credit. We found that exercises with a high model answer view rate have a lower level of post-proficiency attempts. 27% of students solved them post proficiency, compared to almost 50% for other exercises. This is somewhat surprising, as students presumably use an exercise post-proficiency in order to study the material for exams. We might have expected the most difficult exercises to be targets for additional study.

We computed the ratio of correct attempts over number of reset button clicks, and the ratio of all attempts over number of reset button clicks. The correlation between the two ratios was $r^2 = 0.99$. Exercises with lowest ratios (bottom 25%) were related to quicksort, heaps, shellsort, and binary trees topics. When using number of model answer views and use of reset button as measures of exercise difficulty, we found that the hardest exercises are related to the topics of heaps and quicksort. These exercises have higher use of model answers, higher exercise reset rates, and lower levels of post-proficiency attempts compared to other exercises. We note that proficiency exercises cover only algorithm mechanics, and so do not test students on more theoretical concepts. Thus, this analysis is only comparing the relative difficulty

of understanding the mechanics of various algorithms, and so does not address the question of the relative difficulty of algorithm analysis versus algorithm mechanics.

4. INSTRUCTOR SURVEY RESULTS

To validate our process, we compared the results of automated analysis with opinions of course instructors. To that end, we distributed a survey to the CS education community via the SIGCSE mailing list. We asked respondents: (i) how long they have been teaching a post-CS2 course on Data Structures and Algorithms; (ii) what topics from such a course are the most difficult for students to understand; and (iii) what topics from such a course are the most difficult to teach. We received 23 responses with a mean teaching career of 16 years (median 15 years). Since a concept can be defined using different terms, we grouped answers that we considered to refer to the same topic. The result was 12 topics considered most difficult for students to understand, and 8 topics most difficult to teach. Table 2 shows the top 6 difficult topics to learn and to teach. Among the top topics considered hard for students, only trees and heaps are not also present in the list of hard topics to teach.

Table 2: Summary of survey responses

Topic	**N**	**%**
Most difficult topics for students		
Dynamic programming	7	18
Algorithm analysis	6	15
OOP & Design	6	15
Recursion	4	10
Trees, Heaps	3	7
Proofs	3	7
Most difficult topics to teach		
Complex algorithms	8	30
OOP & Design	4	15
Proofs	4	15
Algorithm analysis	3	11
Recursion	3	11
Dynamic programming	2	7

Dynamic programming had the most votes as difficult for students, but we note that most CS3 courses to not cover this in depth. Algorithm analysis received the next highest number of votes. Our IRT and log analyses also identify algorithm analysis as a hard topic for students. Instructors mentioned students' lack of proficiency in mathematics as a major reason why algorithm analysis proves hard. Instructors wrote "mathematical sophistication is the issue here", and "because students are afraid of math". Our analysis of use of trial-and-error also revealed that students are not at ease with mathematics topics. To explain why algorithms analysis is hard to teach, one instructor wrote "I still do not have good instructional material". That reason was also used for other topics like graphs and design. Heaps is another topic that was identified as hard both by our analysis and by instructors. In general, the survey responses correspond fairly well to our automated process.

5. ALGORITHM ANALYSIS IS HARD

Our analysis shows that exercises related to algorithm analysis are harder than exercises covering algorithm mechanics. It also reveals that students might have some difficulty with heaps and quicksort. Algorithm analysis is of particular interest since a main goal of CS3 is to teach students how to analyze algorithms, in order to design efficient software solutions. That is why algorithm analysis sections are present in almost all topics covered in the course. Careful analysis of the data logs reveals certain behaviors by students that could explain why students struggle with these concepts.

5.1 Not spending enough time

We analyzed interaction logs from use of OpenDSA at three universities (Virginia Tech, University of Texas El Paso, and University of Florida). Table 3 shows estimated reading time for the algorithm analysis material from three sorting modules (Insertionsort, Mergesort, and Quicksort). More than 74% of students spent less than one minute on the analysis material for each of the three modules. Based on this result, we believe that most of the students are not reading the analysis material.

Table 3: Time reading algorithm analysis material

University	Module	N	μ(sec)	% < 1 min
VT	Insertionsort	98	63.57	74.48
	Mergesort	96	39.79	78.12
	Quicksort	92	64.71	73.91
UTEP	Insertionsort	26	49.84	80.76
	Mergesort	22	41.45	77.27
	Quicksort	16	16.18	93.75
Florida	Insertionsort	53	40.39	84.90
	Mergesort	44	18.63	95.45
	Quicksort	39	26.12	92.30
All	Insertionsort	177	54.6	78.52
	Quicksort	147	49.2	80.94

86% of students responding to a survey indicated that it is easier for them to understand how an algorithm works than to analyze the running time for that algorithm. Quotes include: *"determining asymptotic running time because it is harder to visualize and less intuitive"*, *"Complexities are confusing and math-like"*, *"I think understanding how an algorithm work is easy. It is the style of presentation"*, *"How the algs work. It is dependent on material, also abstract stuff is harder for me to understand"*.

78% of students who are more comfortable with dynamics attributed this to the material, as algorithm analysis is abstract and requires familiarity with mathematical notations. The other 22% attributed this to how concepts are presented in OpenDSA (dynamics are presented using visualizations, analysis is presented mostly through text). Quotes regarding the usefulness of OpenDSA's algorithm analysis content include: *"Not any more useful than any other book"*, *"Not as much as learning the algorithms themselves, but I felt it was as useful as any resource could be on the topic"*, *"Yes, but not as much as understanding the algorithms"*, *"It could have been more interactive with showing why the analysis was the way that it was"*, *"I found it much more useful on Data structures. Algorithm analysis doesn't benefit quite as much from animations"*, *"It was very detailed and kind of hard to follow"*, *"I'd like there to be more visuals for analysis"*. Clearly respondents did not find the OpenDSA material on algorithm analysis different from other textbooks on that topic. This is not what they expected from OpenDSA, whose goal is to present content interactively.

5.2 Content presentation not engaging

When students were asked to provide suggestions for improving presentation of the analysis material in OpenDSA, most indicated they were expecting a more interactive presentation in the form of visualizations. Quotes include: *"Visualizations definitely help."*, *"I think making the clickthrough pictures into actual animations would be nice"*, *"more animation, the visualizations are great!"*, *"more visualizations is always good"*, *"Visualizations always help :)"*, *"visualizations showing each step of analysis would help"*, *"an animation will make a much bigger difference."*

6. CONCLUSION AND FUTURE WORK

Educational resources are rapidly moving online. As eTextbooks and interactive exercises become more prevalent, techniques to automatically discover the most difficult topics for students will become increasingly important. Doing so allows both instructors and designers of instructional content to focus their resources on the most difficult topics. Perhaps resolving the difficulty might be as simple as fixing a buggy exercise. But more generally, we find that specific concepts are truly hard. By examining the topic in detail, including its method of presentation, we might uncover better approaches to instruction, leading to better outcomes.

To illustrate, we are working on addressing the issues raised by students regarding the lack of visual presentation for algorithm analysis material in OpenDSA. Inspired by the concept of visual proofs [12], a set of Algorithm Analysis Visualizations (AAVs) were implemented for OpenDSA sorting modules [8]. We have collected preliminary data with two small classes using the sorting analysis visualizations. Summary results were collected for two modules teaching Insertion Sort and Quicksort. A Kruskal Wallis tests showed a significant difference ($p < 0.01$) between the time spent for text versus visualizations for these two modules. This indicates that students spend more time on the material when presented as visualizations. Having proved the value of the concept, we will continue to expand on this approach.

7. ACKNOWLEDGMENTS

We gratefully acknowledge the support of the National Science Foundation under Grants DUE-1139861, IIS-1258571, and DUE-1432008.

8. REFERENCES

[1] R. Baker, A. Corbett, and K. Koedinger. Detecting student misuse of intelligent tutoring systems. In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, pages 531–540, 2004.

[2] D. Barker-Plummer, R. Cox, and R. Dale. Student translations of natural language into logic: The grade grinder corpus release 1.0. In *Proceedings of the 4th international conference on educational data mining*, pages 51–60, 2011.

[3] M. Berges and P. Hubwieser. Evaluation of source code with item response theory. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '15, pages 51–56, 2015.

[4] P. Brusilovsky, J. Grady, M. Spring, and C.-H. Lee. What should be visualized?: Faculty perception of priority topics for program visualization. *SIGCSE Bulletin*, 38(2), June 2006.

[5] N. Dale. Content and emphasis in CS1. *SIGCSE Bulletin*, 37(4):69–73, Dec. 2005.

[6] N. B. Dale. Most difficult topics in CS1: Results of an online survey of educators. *SIGCSE Bulletin*, 38(2):49–53, June 2006.

[7] F. Drasgow and C. L. Hulin. Item response theory. *Handbook of industrial and organizational psychology*, 1:577–636, 1990.

[8] M. F. Farghally, E. Fouh, S. Hamouda, K. H. Koh, and C. A. Shaffer. Visualizing algorithm analysis topics. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, page 687, 2016.

[9] E. Fouh, D. A. Breakiron, S. Hamouda, M. Farghally, and C. A. Shaffer. Exploring students learning behavior with an interactive etextbook in computer science courses. *Computers in Human Behavior*, pages 478–485, December 2014.

[10] E. Fouh, V. Karavirta, D. A. Breakiron, S. Hamouda, S. Hall, T. L. Naps, and C. A. Shaffer. Design and architecture of an interactive etextbook–The OpenDSA system. *Science of Computer Programming*, 88:22–40, 2014.

[11] K. Goldman, P. Gross, C. Heeren, G. L. Herman, L. Kaczmarczyk, M. C. Loui, and C. Zilles. Setting the scope of concept inventories for introductory computing subjects. *Transactions on Computing Education*, 10(2):5:1–5:29, June 2010.

[12] M. T. Goodrich and R. Tamassia. Teaching the analysis of algorithms with visual proofs. In *SIGCSE Bulletin*, volume 30, pages 207–211, 1998.

[13] R. K. Hambleton and L. L. Cook. Latent trait models and their use in the analysis of educational test data. *J. of Educational Measurement*, 14(2):75–96, 1977.

[14] P. Jarušek and R. Pelánek. Analysis of a simple model of problem solving times. In S. Cerri, W. Clancey, G. Papadourakis, and K. Panourgia, editors, *Intelligent Tutoring Systems*, volume 7315 of *LNCS*, pages 379–388. Springer, 2012.

[15] L. Malmi, V. Karavirta, A. Korhonen, J. Nikander, O. Seppälä, and P. Silvasti. Visual algorithm simulation exercise system with automatic assessment: TRAKLA2. *Informatics in Education*, 3(2):267–288, September 2004.

[16] G. Rasch. *Probabilistic models for some intelligence and attainment tests.* Danmarks Pædagogiske Institut, 1960.

[17] G. Ravi and S. Sosnovsky. Exercise difficulty calibration based on student log mining. In *Proceedings of DAILE: Workshop on Data Analysis and Interpretation for Learning Environments*, 2013.

[18] L. A. Sudol and C. Studer. Analyzing test items: Using item response theory to validate assessments. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, SIGCSE '10, pages 436–440, 2010.

[19] W. J. van der Linden and R. K. Hambleton. *Handbook of modern item response theory.* Springer, 2013.

Acting the Same Differently: A Cross-Course Comparison of User Behavior in MOOCs

Ben Gelman
Dept. of Computer Science
George Mason University
bgelman@gmu.edu

Matt Revelle
Dept. of Computer Science
George Mason University
revelle@cs.gmu.edu

Carlotta Domeniconi
Dept. of Computer Science
George Mason University
carlotta@cs.gmu.edu

Aditya Johri
Dept. of Computer Science
George Mason University
johri@gmu.edu

Kalyan Veeramachaneni
CSAIL,MIT
Cambridge, MA, USA
kalyan@csail.mit.edu

ABSTRACT

Recent studies of MOOCs demonstrate their ability to reach a large number of users, but also caution against the high rate of dropout. Some have looked closely at MOOC participation in order to better understand how and when users start to disengage, and, if they remain engaged, in what activities they participate. Most of this prior work relies heavily on descriptive statistics or clustering methodologies to highlight basic user participation characteristics. In this paper, we adapt NMF to provide a multi-dimensional view of user participation. We use log data to create a bottom-up understanding of user participation, and identify five basic behaviors associated with participants' use of content and their engagement with assessment. Furthermore, we do a cross-course analysis across four courses and find that these five behaviors are present in all courses. Interestingly, users' participation patterns - how they engage in these five behaviors - vary across courses even when the course topics are similar. Our methodology can be applied to other data sets, and findings from this work can assist in interventions to help users successfully accomplish their learning goals.

Keywords

MOOCs, Participant Behavior, NMF, Comparative Analysis

1. INTRODUCTION

As Massive Open Online Courses (MOOCs) grow in popularity, and offer an increasing variety of subjects across multiple platforms, there has been significant interest in MOOC users' participation patterns. Extremely low user completion rates [6] have motivated examinations and studies of MOOC behavior that aim to ascertain whether changes in pedagogy can improve completion outcomes, or if every incoming class contains a cohort of users that had no intention to complete.

We were motivated by this recent work to attempt to better understand MOOC users' behavioral patterns, and the evolution of participation over time and across courses. In this paper, we analyze data from four MOOC courses across three axes (*learners*, *time*, and *courses*), choosing methods that link behaviors and patterns across these three dimensions. Utilizing the rich features developed to characterize learners' weekly interactions, we adapt non-negative matrix factorization (NMF) [5] to study the importance of these features and the behavior of users over time [2].

Several factors make NMF particularly well-suited for this type of analysis. The non-negativity constraint helps to identify distinct but additive latent factors. In other words, we are able to learn user behaviors in terms of evolving parts due to NMF's additive latent factors and our temporal adaptation (linking behaviors across weeks).Through this study, we make the following unique contributions: 1) We identify behavioral patterns of users that are consistent across multiple MOOCs; 2) We demonstrate how these behaviors vary across different courses; and 3) We demonstrate the feasibility of a framework that can be applied across similar multi-dimensional datasets.

2. RELATED WORK

Several studies of MOOCs highlight low completion rates [13]. The University of Edinburgh launched six MOOCs on the Coursera platform in January 2013 [7]. Evaluations revealed that, of the 309,682 learners initially enrolled, 123,816 (about 40%) accessed the course sites during the first week ('active learners'), and 90,120 (about 29%) engaged with course content. Over the duration of the course, the number of active participants rose to 165,158 (53%). As a gauge of persistence, 36,266 learners (nearly 12%) engaged with week 5 assessments. This represented 29% of initial active learners (although individual numbers for each of the six courses ranged from 7% to 59%). In addition, 34,850 people (roughly 11% of those who enrolled) achieved a statement of accomplishment for reaching a percentage-based benchmark of course completion.

Similarly, when Duke University ran a Bioelectricity MOOC in 2012 [15], 12,175 students initially registered. Only 313 participants (2.6%) achieved a statement of accomplish-

ment. Learner feedback suggested three specific reasons for failure to complete [15]. [8] provides a compilation of available data on MOOC completion. Further analysis of the data shows that, of the 61 courses hosted by Coursera, the average completion rate was just over 6%. This combination of MOOCs' enormous popularity and extremely low completion rate has attracted significant interest.

[17] used a classification method that identifies a small number of longitudinal engagement trajectories in MOOCs. This classifier consistently identifies four prototypical trajectories of engagement: (1) *Completing*, (2) *Auditing*, (3) *Disengaging*, (4) *Sampling*. To decide these engagement patterns, the authors used a number of *binary* variables to determine whether a student accessed a resource or attempted a problem. In contrast, we begin to extract a number of richer descriptors about the students' interaction with the online learning platform.

[9] divides participants into five profiles: no-shows (those who register but never log in); observers (those who log in but do not take assessments); drop-ins (those who participatebut do not attempt to complete the entire course); passive (those seeing the course as content to consume); and active (those participating in all the activities and enriching the course). Similarly, [16] distinguishes five groups of people depending on their level of participation in the MOOC forum: inactive (those that do not visit the forum); passive (those that just consume information); reacting (those that add further aspects to existing questions); acting (those that post questions and lead discussions); and supervising/supporting (those that lead discussions and summarize gained insights).

3. DATA

Our study utilizes four courses, including 6.002x (Fall 2012 and Spring 2013): Circuits and Electronics, 2.01x (Spring 2013): Elements of Structures, 3.091x (Spring 2013): Introduction to Solid State Chemistry. After filtering out learners who had no browsing events for the duration of the courses, the course sizes are 17379, 6339, 5597 and 8870 users, respectively. The course durations are all set to 14 weeks. Using the scripts from the MOOCdb project, we are able to extract 21 features. Table 1 shows the feature numbers and descriptions.

Figure 1 presents the course sizes dynamically. The count of active users for any week is given by the sum of users that have at least one non-zero feature in that week. The count of inactive users is the sum of users that have all-zero feature values in the current week, but had been active in a prior week. New users are those whose first non-zero feature is in the current week. The dropout value is the number of students who are inactive this week and will be inactive for all future weeks.

Because some features are complex and not fully explained by their feature names, we will expand their definitions here. Each feature is computed using the data collected in a week, and generates a single value, so if there are 14 weeks in a course, a user's feature vector will contain 14 values per feature.

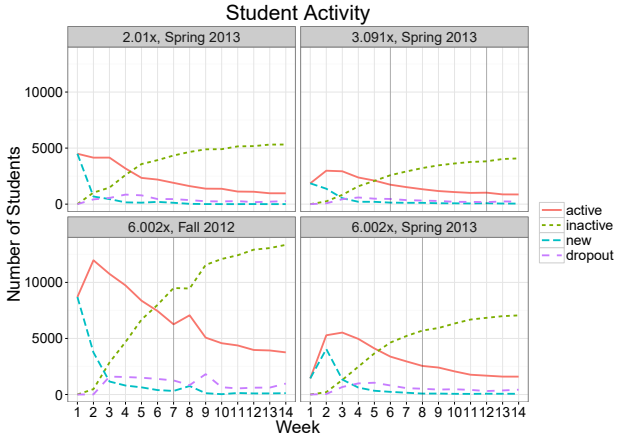


Figure 1: Student activity statuses over time for each class. Vertical lines denote midterm exams and quizzes.

Time spent: Feature 1 sums a user's total time spent on any and all events in the course. Feature 11 is the single longest time spent on any single resource (book, wiki, lecture videos, etc). Feature 12 is the time specifically spent on lectures, and feature 13 is the time spent on the course wiki.

Homework participation: Feature 4 is the count of all unique problems a learner attempted [1]. Feature 5 is the count of all attempts, including multiple tries at the same problem. Feature 6 is the count of all problems that the learner got correct (grade 1). Feature 7 is the average number of attempts per problem. Feature 18 counts all correct attempts, in order to identify users that correctly solve the same problem multiple times.

Ratio-based features: Feature 8 measures the total time spent on the course per correct problem by dividing features 1 and 6. Feature 9 divides the number of attempts (feature 5) by the number of correct problems (feature 6). Feature 19 divides total attempts (feature 5) by non-distinct correct attempts (feature 18).

Difference-based features: Features 14-17 represent the change in features 2, 7, 8, and 9, respectively. This is computed by taking the respective feature's value for the current week, subtracting the previous week, and then normalizing the result.

Regularity and procrastination: Feature 10 tells us how spread out a student's schedule is over the week by presenting the variance of his or her event timestamps. Feature 20 computes the average amount of time the user submits before the deadline (a zero value means an on-time submission, while a higher value means the word was submitted earlier). Finally, feature 21 calculates the standard deviation in working hours throughout the day—if the student starts work around the same time every day, the feature value will be low.

Feature extraction allows us to represent learners as a set of multiple time series. A learner's basic actions are collected and summarized into the 21 interpretive features on a weekly

Table 1: Students' features.

	Features' Names
1	sum_observed_events_duration
2	number_of_forum_posts
3	average_length_of_forum_posts
4	distinct_attempts
5	number_of_attempts
6	distinct_problems_correct
7	average_number_of_attempts
8	sum_observed_events_duration_per_correct_problem
9	number_problem_attempted_per_correct_problem
10	observed_event_timestamp_variance
11	max_duration_resources
12	sum_observed_events_lecture
13	sum_observed_events_wiki
14	difference_feature_2
15	difference_feature_7
16	difference_feature_8
17	difference_feature_9
18	attempts_correct
19	percent_correct_submissions
20	average_predeadline_submission_time
21	std_hours_working

basis. Because learners are represented as a set of features with per-week, aggregate values, time is a dimension of our data set.

4. METHODOLOGY

Uncovering the behaviors of MOOC students requires simultaneously finding interaction patterns (behaviors) across a large number of students and permitting individual students to exhibit multiple behaviors. Since we assume student interactions may be the result of multiple behaviors, we choose to use a decomposition method (NMF) which results in a parts-based representation of student interactions. Students may exhibit multiple behaviors and their behaviors may change over time.

Step 1: Apply NMF Given a three dimensional vector representation of the student feature data with w weeks, f features, and n users, we construct the tensor A_{ijk}. We begin by applying non-negative matrix factorization to each feature-user matrix A_i for $i = [1...w]$. We use a standard implementation [14] with NNDSVD [3] for initialization of the basis matrix and Frobenius cost function. The rank parameter, r, is set to six, which is selected through approximation.

$$\mathbf{A_i} = \mathbb{B}_i\mathbb{C}_i \tag{1}$$

The results of factorizing A_i are B_i and C_i, the *basis* and *coefficient* matrices, respectively. The dimensions of B_i are $f \times r$ and the dimensions of C_i are $r \times n$.

Each of the r column vectors in B_i contain f values that essentially describe the importance of each feature to the given column vector. In our data, we use the set of important features in each basis vector to describe a behavior. In matrix $C_i^\intercal$, there are r column vectors that contain n coefficient values, one for each user. The m^{th} column vector's coefficient values in $C_i^\intercal$ describes how closely users associate with the m^{th} basis vector in B_i. Because every user has r coefficient values, it is possible for a user to identify with multiple basis vectors. This is significantly different than hard clustering approaches such as K-means, where groups are mutually exclusive.

Step 2: Alignment After performing the matrix factorization on each week, we have w basis matrices and w coefficient matrices. To identify persistent basis vectors and patterns, we must connect the results over time. There is no guarantee the order of the basis vectors is consistent over all weeks because the basis matrices are produced by independent executions of NMF. To achieve this, we first compute the cosine similarity using Equation (2) between two consecutive basis vectors. In other words, for each of the r basis vectors in week i, we compute the cosine similarity to all basis vectors in week $i+1$, resulting in r^2 computations. Ultimately, there are $(w-1)r^2$ similarity computations.[1]

$$\mathrm{Sim}(u, v) = \frac{u \cdot v}{||u||_2||v||_2} \tag{2}$$

By examining the distribution of cosine similarity values, an alignment threshold may be selected. For our data, a threshold value of 0.95 was chosen to identify matching basis vectors between weeks. We found that after the first week, all basis vectors uniquely match *one and only one* basis vector in the consecutive week when a threshold of ≥ 0.95 is used. This phenomenon occurred for all four courses we used in our experiments. Although basis matrices for each week are estimated independently, we find five basis vectors which persist over time and occur in all the classes.

Step 3: Normalize and define behaviors The aligned, per-week basis vectors are normalized. We then average these aligned-normalized vectors into a single, representative *behavioral* vector. Having a single, normalized vector permits a semantic interpretation of the behavior based on relative feature values. By identifying the most important features (the ones with the largest values) in each *behavioral* vector, we are able to label the vectors by the interaction pattern they best represent.

Step 4: Coefficient analysis

Every student's interaction attributes may be approximated using a weighted mixture of the discovered behavior vectors. These weights (coefficients) can be considered to define a soft-membership of a student to a behavior.

In order to decide if a user belongs to a behavior, we threshold the distribution of the coefficient values per

[1]We choose cosine similarity because it is a measure of angular similarity between two vectors. Thus, two basis vectors whose only nonzero entry is feature j will be extremely similar. This is valuable for aligning basis vectors whose distributions of features are similar.

week and per behavioral vector (or basis). This means that the algorithm will generate $r \times w$ thresholds. The thresholding algorithm takes the entire range of coefficient values per vector and limits the range of values to the top x%. The threshold (top x%) is a parameter. This means that if the range of coefficient values for a behavior is 0-100, then selecting a threshold of 0.85 will only consider users with coefficient values of 85-100 to be exhibiting that behavior. There is an additional minimum size parameter s that adjusts for a skewed distribution where a few users have significantly higher coefficient values that any other users. This skewed distribution causes the top x% of coefficient values to only include these few users. If the number of users within the top x% is less than the s, then the users will be saved, and the threshold computation will be repeated without them. For our data, we use a threshold of 0.85 with a minimum size parameter of 30.

We assign behaviors to students for each week using the data-derived thresholds. By tracking the set of behaviors across weeks, we generate a transition diagram that presents the number of students exhibiting each behavior over each week and the migration of users between various behaviors. The transition diagram allows us to understand the evolution of user behavior as a course progresses.

5. BASIS MATRIX RESULTS

The resulting basis matrices for 6.002x (Fall 2012) exhibit eight unique behaviors. Tables 2 and 3 numerically summarize behaviors for week one and the average of the other weeks, respectively. Because the first week manifests two unique behaviors, namely *introduction* and *sampling*, it is kept separate. From the second week onwards, all behaviors are persistent (at least 95% cosine similarity). This allows us to average weeks two through 14 in Table 3.

Basis vector one is dominated by feature 11 (max_duration_resources), which is the duration of the longest observed event this week. This vector represents a *deep* behavior, because the associated students must have spent a long time on a single resource.

Basis vector two is primarily decided by feature 10 (observed_event_timestamp_variance). Because this feature tells us how spread out the student's schedule is over the week, this vector describes a *consistent* behavior. Having a high timestamp variance requires users to log in multiple times a week.

Basis vector three is primarily decided by feature 21 (std_hours_working), which is the standard deviation in working hours over the day. This could represent a *bursty* behavior—because a user must be active during different times in a day to obtain a high feature value, this could mean that the user has a single prolonged session or multiple, separate sessions.

Two basis vectors exist only in the first week of the course. Basis vector four in Table 2 is decided by feature three (average_length_of_form_posts) and feature two (number_of_form_posts). This supports the idea that users interacted heavily during the opening week of the course. The disappearance of this basis vector, however, tells us that forum interaction in later parts of the course was insignificant in 6.002x fall 2012. For this reason, this basis vector characterizes an *introduction* behavior.

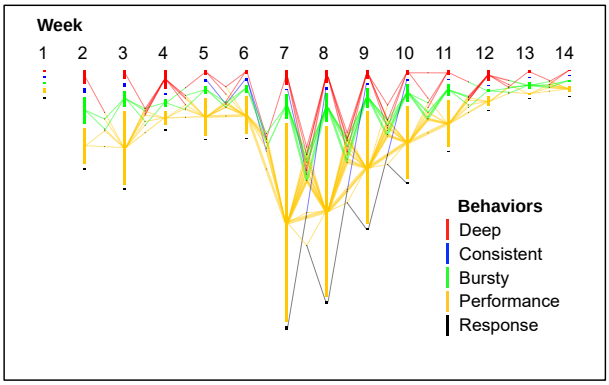
Table 2: Matrix of normalized basis vectors (behaviors) for week 1 (course 6.002x fall 2012). The behaviors *Introduction* and *Sampling* are unique to week 1. Dominant feature values are shown in boldface.

Feature	*Deep*	*Consistent*	*Bursty*	*Introduction*	*Sampling*
1	0.012	0.000	0.001	0.000	0.088
2	0.000	0.000	0.000	**0.137**	0.000
3	0.000	0.000	0.000	**0.862**	0.000
4	0.000	0.000	0.000	0.000	0.000
5	0.000	0.000	0.000	0.000	0.000
6	0.000	0.000	0.000	0.000	0.000
7	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000
10	0.000	**0.988**	0.000	0.000	0.000
11	**0.981**	0.011	0.000	0.001	0.000
12	0.000	0.000	0.000	0.000	**0.665**
13	0.000	0.000	0.000	0.000	0.000
14	0.000	0.000	0.000	0.000	0.000
15	0.000	0.000	0.000	0.000	0.000
16	0.000	0.000	0.000	0.000	0.000
17	0.000	0.000	0.000	0.000	0.000
18	0.000	0.000	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	0.000
20	0.000	0.000	0.000	0.000	0.000
21	0.008	0.000	**0.999**	0.000	**0.248**

Basis vector five in Table 2 is defined by features 12 (sum_observed_events_lecture), 21 (std_hours_working), and 1 (sum_observed_events_duration). This group of features supports the hypothesis that users are browsing through a lot of content during the first week of the course. This may be because users are interested in seeing what lies ahead in the course, or because some users may have joined only to gather information on one particular topic. Thus, basis vector five during the first week expresses a *probing* behavior.

After the first week, two more basis vectors persist. At this point, basis vector four is primarily characterized by feature 19 (percent_correct_submissions). By turning in assignments with high correctness, the corresponding students can be associated with a *performance* behavior. Basis vector five is strongly defined by feature 20 (average_predeadline_submission_time). By turning in assignments long before their deadlines, these students can be associated with an *response* behavior.

When we apply the same analysis to other courses, we see similar behaviors. The average basis matrix tables for 2.01x, 3.091x, and 6.002x are not displayed because they exhibit the same behaviors as table 3 with 95% cosine similarity. It appears that each of these five behaviors— deep, consistent, bursty, performance, and response—appear in all of the courses. The key difference is that 6.002x has two additional behaviors that occur only in the first week. The introduction and sampling behaviors do not appear to be prevalent in the other courses. This could be due to course



(d) 6.002x, Spring 2013

Figure 2: User behavior transitions over time. Vertical bars are numbers of students performing each behavior. Diagonal groups indicate transitions: for example, the transition ⟩ indicates students who were Deep and Bursty and have transitioned to Consistent. Transition thickness is the log of the number of students involved.

repeat occasionally, they only occur for two to three weeks at a time. Thus, we do not infer any transitional motifs from this course.

In 6.002x fall, most user migration occurs through the deep behavior, with a secondary focus on the consistent behavior. A unique circumstance occurs between weeks one and two with the migration of the initially enormous bursty behavior. Besides this, the transitional motifs include each permutation of deep and/or consistent migrating to deep and/or consistent.

In 6.002x spring, most user migration occurs through the performance behavior. Unlike the other courses, there are two more behaviors through which there is significant migration: the deep and bursty behaviors. As a result, we see many more motifs than simply the permutations of the top two behaviors. In the early weeks, migration is heaviest through deep and performance. This means that early on, users are both engaged and performing well. In the middle weeks, during and after the midterm, there is a chaotic shuffle between behaviors as users deal with the course differently. In the later weeks, however, deep migration falls off and users mostly move between bursty and performance. This may suggest that users are capable of finishing their work in a single day or two and achieving high correctness simultaneously. This result could perhaps reflect a decreased difficulty in the later weeks of the course. The occurrence of multiple large behaviors appears to tells us more about the evolution of user behavior.

7. CONCLUSION

In this comparative study of four MOOC courses, we show how users follow five specific behaviors across the courses. We found that although these behaviors are common, their patterns of occurrence vary across courses. Through our multi-dimensional data and our adaptation of NMF, the results reveal in great detail the differences in behavior over time between the courses. Because our method analyzes behavior at every step of the MOOC experience, our work can improve the learning experience for all users, not just those that plan to finish the course. For future work, we can expand the purposes of user behavior trajectories by using Markov modeling for prediction. We can add newer, more descriptive features in addition to running the analysis with a higher rank in order to discover possible alternative behaviors. If course outcomes and assessment information are available, we can combine these with the dynamic behavioral motifs to better understand the underlying processes that fuel behavioral changes.

8. REFERENCES

[1] Veeramachaneni, Kalyan, Halawa, Sherif, Dernoncourt, Franck, O'Reilly, Una-May, Taylor, Colin, and Do, Chuong. Moocdb: Developing standards and systems to support mooc data science. arXiv preprint arXiv:1406.2015, 2014a.

[2] Sra, Suvrit, and Inderjit S. Dhillon. "Generalized nonnegative matrix approximations with Bregman divergences." Advances in neural information processing systems. 2005.

[3] Boutsidis, Christos, and Efstratios Gallopoulos. "SVD based initialization: A head start for nonnegative matrix factorization." Pattern Recognition 41.4 (2008): 1350-1362.

[4] Swinson, Christina J. "Mathematica." CJs Blog of Miscellanies and Accelerator Physics. N.p., 21 Jan. 2011. Web. 05 Mar. 2016.

[5] Lee, Daniel D., and H. Sebastian Seung. "Learning the parts of objects by non-negative matrix factorization." Nature 401.6755 (1999): 788-791.

[6] Breslow, Lori, et al. "Studying learning in the worldwide classroom: Research into edX's first MOOC." Research & Practice in Assessment 8 (2013).

[7] MOOCs@Edinburgh Group. MOOCs@Edinburgh (2013): Report#1, Available at: http://hdl.handle.net/1842/6683 [Accessed: 20/01/14].

[8] Jordan, K. (2013). MOOC Completion Rates: The Data, Available at: http://www.katyjordan.com/MOOCproject.html [Accessed: 18/02/14].

[9] Hill, P. (2013). The Most Thorough Summary (to date) of MOOC Completion Rates|e-Literate. e-Literate blog. Retrieved June 10, 2013, from http://mfeldstein.com/the-most-thorough-summary-to-date-of-mooc-completion-rates/

[10] Liyanagunawardena, T. R., Adams, A. A. & Williams, S. A. (2013). MOOCs: a systematic study of the published literature 2008–2012. The International Review of Research in Open and Distance Learning, 14, 3, 202–227.

[11] Ebben, M. & Murphy, J. S. (2014). Unpacking MOOC scholarly discourse: a review of nascent MOOC scholarship. Learning, Media and Technology, 39, 3, 1–18. doi: 10.1080/17439884.2013.878352

[12] EDUCAUSE (2012). What Campus Leaders Need to Know About MOOCs. EDUCAUSE BRIEFS. Retrieved June 10, 2013, from http://www.educause.edu/library/resources/what-campus-leaders-need-know-about-moocs

[13] Onah, Daniel F. O., Sinclair, Jane and Boyatt, Russell (2014) Dropout rates of massive open online courses : behavioural patterns. In: 6th International Conference on

[14] Pedregosa et al., Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011. Education and New Learning Technologies, Barcelona, Spain, 7-9 Jul 2014. Published in: EDULEARN14 Proceedings pp. 5825-5834.

[15] Belanger, Y. (2013). Bioelectricity : A Quantitative Approach, Available at http://dukespace.lib.duke.edu/dspace/bitstream/handle/10161/6216/Duke_Bioelectricity_MOO C_Fall2012.pdf

[16] F. Gruenwald, E. Mazandarani, C. Meinel, R. Teusner, M. Totschnig, and C. Willems, "openHPI-a Case-Study on the emergence of two learning communities," in Proc. IEEE Global Eng. Edu. Conf., 2013, pp. 13–15.

[17] Kizilcec, R. F., Piech, C., & Schneider, E. (2013, April). Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In Proceedings of the third international conference on learning analytics and knowledge (pp. 170-179). ACM.

Collaborative Problem Solving Skills versus Collaboration Outcomes: Findings from Statistical Analysis and Data Mining

Jiangang Hao
Educational Testing Service
ETS Rosedale Road, MS 02-T
Princeton, NJ 08541
jhao@ets.org

Lei Liu
Educational Testing Service
ETS Rosedale Road
Princeton, NJ 08541
lliu001@ets.org

Alina A von Davier
Educational Testing Service
ETS Rosedale Road
Princeton, NJ 08541
avondavier@ets.org

Patrick Kyllonen
Educational Testing Service
ETS rosedale Road
Princeton, NJ 08541
pkyllonen@ets.org

Christopher Kitchen
Educational Testing Service
ETS Rosedale Road
Princeton, NJ 08541
ckitchen@ets.org

ABSTRACT

With the aid of educational data mining and statistical analysis, we investigate the relationship between collaboration outcomes and collaborative problem solving (CPS) skills exhibited during the collaboration process. We found that negotiation skill contributes positively to the collaboration outcomes while purely sharing information does the opposite.

Keywords

collaborative problem solving, simulation-based assessment, random forest

1. INTRODUCTION

Collaborative problem solving (CPS) is widely considered as one of the critical skills for academic and career success in the 21st century [9]. However, assessing CPS, particularly in a large-scale and standardized way, is very challenging, as one must take into account the forms of collaboration, the size of teams, and assessment contexts. Among the existing studies on assessing CPS, most of them are not designed from the perspective of a standardized assessment, but more from the perspective of revealing some important aspects of CPS [6, 16, 5, 22]. A recent review can be found in [21]. The first large-scale and standardized assessment for CPS was the international Assessment and Teaching of 21st century skills project (ATC21S) carried out by Griffin and colleagues [9, 4]. In this assessment, two students collaborate via text chat to solve computer-based CPS tasks and their communications as well as some other features (such as the response time) were coded automatically according to a CPS framework [1]. Another large-scale assessment for CPS was carried out by the Programme for International Student Assessment (PISA) in its sixth survey in 2015 [17]. In this assessment, students collaborate with different number of virtual partners (avatars) on a set of computer-based collaborative tasks and they communicate with their virtual partners by choosing from a list of predefined texts. Both ATC21S and PISA 2015 consider the CPS as skills across different domains and the tasks used in their assessments are not confined into a specific domain.

In this paper, we report our findings on the relationship between the CPS skills and the collaboration outcomes in the domain of science, as we think CPS is more likely to be domain dependent. We developed a simulation-based task, in which two participants collaborate via text chat to complete a set of questions and activities on volcanoes [10]. We choose a simulation-based task because it provides students with opportunities to demonstrate proficiencies in complex interactive environments that traditional assessment formats cannot afford [14], which is especially suitable for measuring the complex skills such as CPS.

In the simulation task, for each item, we ask each member of a dyadic team to respond individually first (initial response). Then, after collaboration, each of them will be given a chance to submit a revised response. The difference between the initial and revised responses directly encodes the effect due to collaboration. Based on the data collected using Amazon Mechanical Turk, we introduce two variables, "number of changes" and "score change", to characterize the collaboration outcomes. The "number of changes" is the total number of attempts by the team members to change the initial responses after the collaboration. Some of the attempts change the responses from correct to incorrect while some change the responses from incorrect to correct. This number reflects the willingness to make a change after the collaboration. On the other hand, the "score change" is the sum of the score changes between the initial and revised responses, which quantifies the results of the changes. Based on these two variables, we classify the teams into "effective

collaboration" (e.g., teams that have positive "score change") and "ineffective collaboration" (e.g., teams that have negative "score change" or zero "number of changes").

In addition to quantifying the collaboration outcomes, we introduced a "CPS profile" to characterize the CPS skills exhibited by each team during the collaboration process. The CPS profile is defined as the frequency distribution of CPS skills (unigram) and the consecutive CPS skill pairs (bigram). Random forest classification analysis [12, 3] is used to analyze the relationship between collaboration outcomes and the CPS skills. Random forest is a decision tree-based binary classifier, with increased robustness by using multiple trees rather than a single tree. It is mainly used as a classifier to map the features (independent variables) to labels (dependent variables). When training a random forest classifier, the relative importance of the feature variables for determining the labels can be obtained as a by-product. In our case, the feature variables are the CPS profile and the labels are the two classes of collaboration outcomes, e.g., effective and ineffective collaborations. By training a random forest classifier on the data, we found that negotiation skill is more important for a successful collaboration outcome.

2. METHOD

2.1 Assessment Instruments

We designed a research study to explore the relationship between CPS skills and the collaboration outcomes. In this large-scale study, we focused on the domain of science and limited the number of members of each team to two. We used text chat as the collaboration medium. There were two major assessment instruments:1) A standalone test for general science knowledge consisting of 37 multiple-choice items adapted from the Scientific Literacy Measurement (SLiM) instrument [18]; 2) A web-based collaborative simulation task on volcanoes that require two participants collaborate to complete.

The simulation task was modified from an existing simulation, Volcano Trialogue [23]. In this simulation task, two participants worked together via text chat to complete the tasks. All of the turn-by-turn conversations and time-stamped responses to the questions were recorded in a carefully designed log file [11]. These conversations were used to measure CPS skills, while the responses to the in-simulation science items were used to measure science inquiry skills [23]. Figure 1 shows screenshot of the simulation task.

To capture the evidence for the outcomes of the collaboration, we designed a four-step response procedure for each item in the task: 1) Each participant was prompted to respond the item individually before any collaboration; 2) Each participant was prompted to discuss the item with her partner; 3) Each participant was prompted to revise her initial response if she wanted; 4) A representative was randomly chosen to submit a team answer.

In this way, the changes in the responses before and after the collaboration reflect how effective the collaborations were and allow us to probe directly what CPS skills are more important for better collaboration outcomes.

2.2 Participants and Data

Figure 1: Screenshots from the collaborative simulation task.

We collected data through Amazon Mechanical Turk, a crowdsourcing data collection platform [13]. We recruited 1,000 participants with at least one year of college education to take the general science test. Then, they were teamed randomly into dyads to take the collaborative simulation task.

After removing incomplete responses, we had complete responses from 493 dyads. However, a further scrutiny of the data showed that many of the teams started some conversations even before the system prompted them to discuss. This means that they started conversations before or during the period that they are supposed to make initial responses individually. Different teams had nonprompted conversations for a different subset of the items, which complicates the analysis. Of the teams, 82 did not have nonprompted conversatons while the other teams had nonprompted discussions for a varying number of items. We compared the scores of the general science knowledge test for participants from the 82 teams with the scores for the rest of the teams via a two-tailed t-test for independent samples, and the resulting p-value is 0.38. This indicates that participants from the 82 teams are not different in a statistically significant way from the rest of the participants in terms of the general science knowledge. To make our analysis clean, we will stick to the data from this 82 teams throughout this paper.

The data from the simulation task for each team include the responses to the items in the simulation and the text chat communications between the dyads around each item. There are 7 multiple-choice equivalent items. Around each item, there are about 5 turns of conversations.

2.3 Analysis

The focus of this paper is to investigate the relationship between the CPS skills and the collaboration outcomes. As such, our analysis focuses on the responses and communications in the collaborative simulation task.

2.3.1 Scoring and Annotating

Students' responses to the seven multiple-choice equivalent items were scored based on the corresponding scoring rubrics as presentend in [23]. In addition to the outcome response

data, we also applied a CPS framework to annotate the chat communications during the collaboration [15]. This CPS framework was developed based on the findings from computer-supported collaborative learning (CSCL) research [2, 7, 9, 21] and the PISA 2015 Collaborative Problem Solving Framework [17].

The framework outlines the four specific categories of the CPS construct (skills) we would like to focus on: *sharing ideas*, *negotiating ideas*, *regulating problem-solving activities*, and *maintaining communication*. Each of these major categories had some subcategories and the total number of subcategories amounted to 33 and a summary of the coding rubrics can be found in Table 1. All the coding was done at the subcategory level, based on which of the four major categories were assigned at a later point.

Two human raters were trained on the CPS framework, and they double-coded a subset of the discourse data (15% of the data). The unit of analysis was each turn of a conversation, or each conversational utterance. The raters had two training sessions before they started independent coding. In the first session, the author of the CPS framework (the second author) trained both raters on the 33 subcategories of CPS skills using the skills definitions and coding examples for each subcategory. In the second training session, the trainer and two raters coded data from one dyad together to practice the application of specific codes and address issues specific to classifying utterances using the CPS framework. After the training sessions, the two raters independently coded discourse data from about 80 dyads.

We used the unweighted kappa statistic to measure the degree of agreement between the human raters' coding. The unweighted kappa was 0.61 for all 33 subcategories and 0.65 for the four major categories. According to Fleiss and Cohen [8], a kappa value of 0.4 is an acceptable level of agreement for social science experiments.

2.3.2 Quantifying the Collaboration Outcomes

The difference between the revised response and initial response is a direct measure of the collaboration outcomes. If we treat each dyad as the unit of analysis, we need to define variables to quantify the answer changes for each item. We first introduce the "number of changes" (denoted as n) to quantify how many revised responses are different from initial responses from both members of each dyad for each item. The possible values for n are {0, 1, 2}: n is zero when nobody makes any changes, one when only one person makes changes, and two when both members make changes. Next, we introduce "score change" (denoted as s) to quantify the total score changes between the revised response and the initial response from both members of each dyad for each item. The definition of s is the sum of the score difference between initial responses and revised responses for the two members of each dyad. The possible states for s are {-2, -1, 0, 1, 2}. One should note that for the state s = 0, there are two different possibilities. The first is that both members do not change their responses. The second is that one member changes a response from incorrect to correct and the other changes from correct to incorrect. Therefore, to have a complete description of the changes at a dyadic level, we introduce the vector "item collaboration effect" for each item, $\delta_k = (s_k, n_k)$, with δ_k defined at the item level and subscript k denoting the item number. At the task level, we simply sum all items, which gives $\Delta = (S, N)$, where $S = \sum_k s_k$ and $N = \sum_k n_k$. By convention, we use the lowercase n and s to denote the item level changes and the uppercase N and S to denote the task-level changes.

2.3.3 Quantifying the CPS Skills

Each turn-by-turn conversations was classified in one of the four categories of CPS skills (e.g., share ideas, negotiate ideas, regulate problem solving, and maintain communication). We introduce a "CPS profile" as a quantitative representation of the CPS skills of each dyad. The profile was defined by the frequency counts of each of the four CPS-skill categories or their combinations and had two levels, unigram and bigram. The unigram, bigram, or even ngram levels are used in natural language processing to represent text. We borrow this idea here to represent CPS skills and limit us to the unigram and bigram as the frequency count is too low for other ngram. The frequency counts of the different CPS skills were used at the unigram level, while the frequency counts of consecutive pairs of CPS skills in the conversations were used at the bigram level. As such, each dyadic team's communications can be represented by the corresponding CPS profile.

It is worth noting that though we consider only unigram and bigram of the CPS skills, other collaboration-related information can also be appended to the profile. For example, the number of turns, the total number of words, etc. Such a profile is essentially a vector representation of collaboration skills exhibited by each team. The vector nature of this representation allows us to easily calculate "similarity" or "dissimilarity" among the teams, which is the foundation of cluster analysis.

3. FINDINGS

We have introduced two variables, N and S, to quantify the collaboration outcomes. We also introduced the CPS profile to quantify the CPS skills. Now, we investigate the relationship between the CPS skills and the collaboration outcomes.

3.1 Effective versus Ineffective Collaboration

Based on the N and S variables, we define the effective collaboration and ineffective collaboration as follows

- Effective collaboration: $N > 0 \ \cap \ S > 0$.
- Ineffective collaboration: $(N > 0 \ \cap \ S \leq 0) \ \cup \ N = 0$.

We need to point out that the criteria for effective collaboration is not necessarily a fixed one. In the current study, we considered the collaboration as effective as long as at least one member made at least a total net change from incorrect to correct. If nobody in the team made at least one total net correct change, we thought of the collaboration as ineffective. Figure 2 shows how the 82 teams were distributed in the space spanned by S and N.

Table 1: Coding rubric of CPS skills used in this paper was developed based on a review of CSCL research findings [2, 7, 9], and the PISA 2015 Collaborative Problem Solving Framework [17], with a focus on CPS in the domain of science. More details about the CPS framework can be found in [15].

CPS skills	Student performance (subcategories)
Sharing ideas	1. Student gives task-relevant information (e.g., individual response) to the teammate. 2. Student points out a resource to retrieve task-relevant information. 3. Student responds to the teammate's request for task-relevant information.
Negotiating ideas	4. Student expresses agreement with the teammates. 5. Student expresses disagreement with teammates. 6. Student expresses uncertainty of agree or disagree. 7. Student asks the teammate to repeat a statement. 8. Student asks the teammate to clarify a statement. 9. Student rephrases/complete the teammate's statement. 10. Student identifies a conflict in his or her own idea and the teammate's idea. 11. Student uses relevant evidence to point out some gap in the teammate's statement. 12. Student elaborates on his or her own statement. 13. Student changes his or her own idea after listening to the teammate's reasoning
Regulating problem solving	14. Student identify the goal of the conversation. 15. Student suggests the next step for the group to take. 16. Student expresses confusion/frustration or lack of understanding. 17. Student expresses progress in understanding. 18. Student reflects on what the group did. 19. Student expresses what is missing in the teamwork to solve the problem. 20. Student checks on understanding. 21. Student evaluates whether certain group contribution is useful or not for the problem solving. 22. Student shows satisfaction with the group performance. 23. Student points out some gap in a group decision. 24. Student identifies a problem in problem solving.
Maintaining communication	25. Student responds to the teammate's question (using texts and text symbols). 26. Student manages to make the conversation alive (using texts and text symbols, using socially appropriate language). 27. Student waits for the teammate to finish his/her statement before taking turns. 28. Student uses socially appropriate language (e.g., greeting). 29. Student offers help. 30. Student apologizes for unintentional interruption. 31. Student rejects the teammateâĂŹs suggestions without an accountable reason. 32. Student inputs something that does not make sense. 33. Student shows understanding of the teammate's frustration.

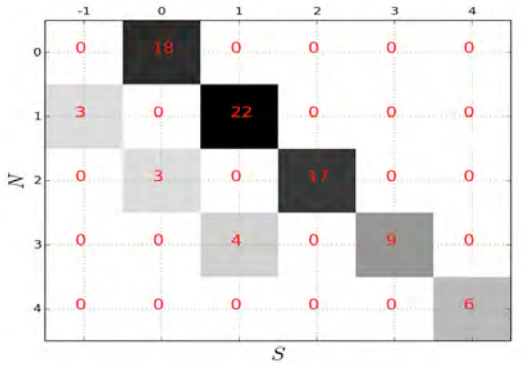


Figure 2: The distribution of the teams in space spanned by N and S.

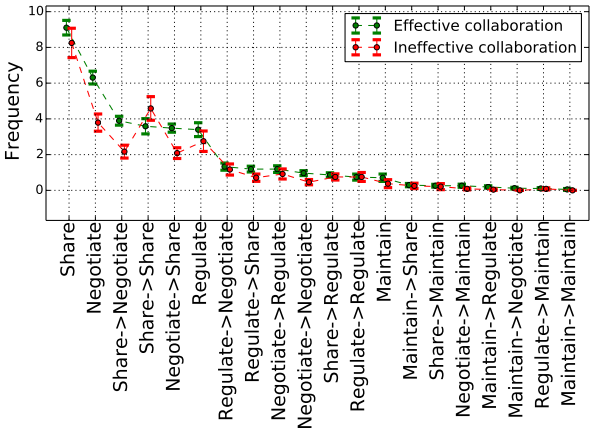


Figure 3: Unigram and bigram profile of CPS skills for the teams corresponding to effective and ineffective collaborations.

Next, we compare the mean CPS profiles of the teams from the effective and ineffective collaborations and the results are shown in Figure 3.

From these results, one can readily see that at the unigram level, the teams with effective collaboration show statistically significantly more negotiating skills than the teams with ineffective collaboration. At the bigram level, teams with effective collaboration exhibited statistically significantly more of the following consecutive CPS skill pairs: share-negotiate, negotiate-share, regulate-share, and negotiate-negotiate. However, the teams with ineffective collaboration showed many more share-share skill pairs.

3.2 Relative Importance of CPS Skills

Figure 3 shows certain CPS skills exhibit more different frequency for effective and ineffective collaborations, which means they have more weight in determining the collaboration outcomes. To get a more quantitative measure of the relative importance of each CPS skills (or skill pairs), we used two methods as follows.

First,we perform a t-test for each of the CPS skills (or skill pairs) for the effective collaboration and ineffective collaboration groups. We use the corresponding p-value to tell which skills or skill pairs show more distinction. The p-value for each component of the CPS profile was shown in Figure 4. If we choose 0.05 as the significance level, negotiate, share-negotiate, negotiate-share and negotiate-negotiate stand out immediately.

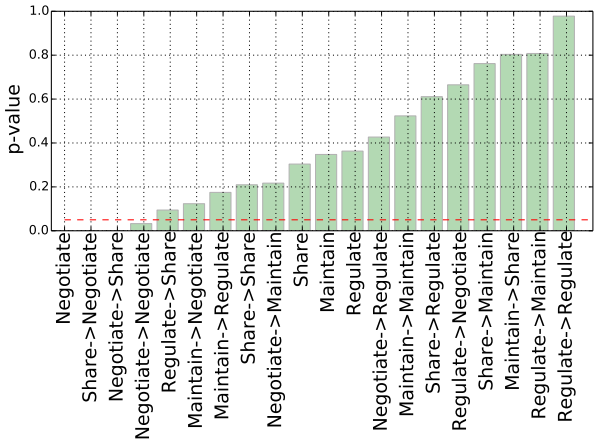


Figure 4: P-value of t-test on the frequency of different CPS skills corresponding to effective and ineffective collaborations. The red horizontal dashed line corresponds to a significant level of 0.05.

A second method we used to find out the relative importance of the CPS skills or skill pairs (feature variables) is random forest classifier [12, 3]. We choose the collaboration outcomes as label variables. During the training of the classifier, a set of decision cuts were made on each feature variable. The relative depth of a feature used as a decision node in a decision tree represents the relative importance of that feature with respect to the predictability of the target labels. Generally speaking, features used at the top level of the decision tree will affect a larger fraction of the sample in terms of the final prediction. Therefore, the expected fraction over the trees in the forest can be used as an estimate of the relative importance of the features. Figure 5 shows the relative importance of the CPS skills and skill pairs based on such an analysis. The results show that negotiation-related skills top the ranking.

The results from these two different analyses converge nicely on that negotiation is a very critical skill for successful collaboration. This finding is consistent with the findings in the literature on knowledge-building discourse [19, 20], as knowledge is often built upon its use and negotiation includes interpretive process of making meaning of exchanged ideas.

4. CONCLUSIONS AND IMPLICATIONS

In this paper, we introduced a CPS profile approach to quantify the CPS skills of each team and found that the negotiation skill at the unigram level is important for better collaboration outcomes. At the bigram level, we found that more negotiation-related skill pairs, such as share-negotiate,

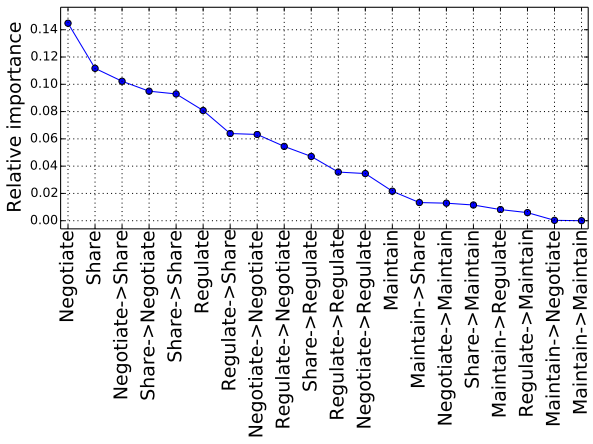


Figure 5: Relative feature importance based on a random forest classifier.

negotiate-share, regulate-share, and negotiate-negotiate, leads to more effective collaboration outcomes. However, purely sharing information with each other (share-share) is associated with poorer collaboration outcomes. This empirical finding may also inform the development of an outcome-oriented scale for CPS skills.

The current study also has limitations. For example, the items in the task are all relatively easy so that there are few turns for each item. There are not many items in the task, which limits the effect of the collaboration outcomes. All these issues will be resolved in our next round of data collection and analysis.

5. ACKNOWLEDGMENTS

Funding for this project is provided by Educational Testing Service through the game, simulation and collaboration initiative.

6. REFERENCES

[1] R. Adams, A. Vista, C. Scoular, N. Awwal, P. Griffin, and E. Care. Automatic coding procedures. *Assessment and teaching of 21st century skills*, 2, 2015.

[2] B. Barron. When smart groups fail. *The journal of the learning sciences*, 12(3):307–359, 2003.

[3] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] E. Care and P. Griffin. An approach to assessment of collaborative problem solving'. *Research & Practice in Technology Enhanced Learning*, 9(3):367–388, 2014.

[5] E. G. Cohen, R. A. Lotan, B. A. Scarloss, and A. R. Arellano. Complex instruction: Equity in cooperative learning classrooms. *Theory into practice*, 38(2):80–86, 1999.

[6] L. A. DeChurch and J. R. Mesmer-Magnus. The cognitive underpinnings of effective teamwork: a meta-analysis. *Journal of Applied Psychology*, 95(1):32, 2010.

[7] P. Dillenbourg and D. Traum. Sharing solutions: Persistence and grounding in multimodal collaborative problem solving. *The Journal of the Learning Sciences*, 15(1):121–151, 2006.

[8] J. L. Fleiss and J. Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 1973.

[9] P. Griffin, B. McGaw, and E. Care. *Assessment and teaching of 21st century skills*. Springer, 2012.

[10] J. Hao, L. Liu, A. von Davier, and P. Kyllonen. Assessing collaborative problem solving with simulation based tasks. *proceeding of 11th international conference on computer supported collaborative learning*, 2015.

[11] J. Hao, L. Smith, R. Mislevy, A. von Davier, and M. Bauer. Taming log files from game and simulation based assessment: data model and data analysis tool. *ETS Research Report*, in press.

[12] T. K. Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995.

[13] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[14] E. Klopfer, S. Osterweil, J. Groff, and J. Haas. Using the technology of today, in the classroom today. *The Education arcade*, 2009.

[15] L. Liu, J. Hao, A. A. von Davier, P. Kyllonen, and D. Zapata-Rivera. A tough nut to crack: Measuring collaborative problem solving. *Handbook of Research on Technology Tools for Real-World Skill Development*, page 344, 2015.

[16] H. F. O'Neil. *Workforce readiness: Competencies and assessment*. Psychology Press, 2014.

[17] Organization for Economic Co-operation and Development [OECD]. Pisa 2015 draft collaborative problem solving assessment framework. *OECD Publishing*, 2013.

[18] C.-J. Rundgren, S.-N. C. Rundgren, Y.-H. Tseng, P.-L. Lin, and C.-Y. Chang. Are you slim? developing an instrument for civic scientific literacy measurement (slim) based on media coverage. *Public Understanding of Science*, 21(6):759–773, 2012.

[19] M. Scardamalia and C. Bereiter. Computer support for knowledge-building communities. *The journal of the learning sciences*, 3(3):265–283, 1994.

[20] G. Stahl. *Group Cognition: Computer Support for Building Collaborative Knowledge (Acting with Technology)*. The MIT Press, 2006.

[21] A. A. Von Davier and P. F. Halpin. Collaborative problem solving and the assessment of cognitive skills: Psychometric considerations. *ETS Research Report Series*, 2013(2):i–36, 2013.

[22] A. W. Woolley, C. F. Chabris, A. Pentland, N. Hashmi, and T. W. Malone. Evidence for a collective intelligence factor in the performance of human groups. *science*, 330(6004):686–688, 2010.

[23] D. Zapata-Rivera, T. Jackson, L. Liu, M. Bertling, M. Vezzu, and I. R. Katz. Assessing science inquiry skills using trialogues. In *Intelligent Tutoring Systems*, pages 625–626. Springer, 2014.

Hint Availability Slows Completion Times in Summer Work

Paul Salvador Inventado,
Peter Scupelli
School of Design
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA, USA
paulsb@andrew.cmu.edu,
pgs@andrew.cmu.edu

Eric G. Van Inwegen,
Korinn S. Ostrow,
Neil Heffernan III
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA, USA
egvaninwegen@wpi.edu,
ksostrow@wpi.edu,
nth@wpi.edu

Jaclyn Ocumpaugh,
Ryan S. Baker,
Stefan Slater,
Mia Almeda
Teachers College, Columbia University
525 W. 120th Street,
New York, NY, USA
jo2424@tc.columbia.edu,
baker2@exchange.tc.columbia.edu,
slater.research@gmail.com,
victoriaalmeda@gmail.com

ABSTRACT
On-demand help in intelligent learning environments is typically linked to better learning, but may lead to longer completion times. This present work provides an analysis of how students interacted with a summer learning assignment when on-demand help was available, compared to when it was not. When hints were available from the start, students were more likely to delay work, compared to students for whom step-wise hints were only available after the third problem. When hints were always available, participants took significantly more time to complete a mastery learning assignment. We interpret this difference in time to complete the assignment as an opportunity to re-engage in productive math learning.

Categories and Subject Descriptors
H1.2 [**Information Systems**]: User/Machine Systems – human factors

General Terms
Measurement, Design, Experimentation, Human Factors

Keywords
Hints, completion time, randomized controlled trial, ASSISTments

1. INTRODUCTION
Help-functions—including on-demand help, contextualized hints, or supplementary learning materials [2]—are a major asset of modern intelligent learning environments. These functions have often been associated with better student learning outcomes ([1][9][25]), but not all help has proven equally effective, and even well-crafted hints may be used ineffectively by students who do not actually need them ([2][20]) . Research has shown cases in which help functions fail [1] and has sought to identify the contexts in which different types of help strategies are most effective ([12][22]).

EDM '16, June 29–July 2, 2016, Raleigh, NC, USA.

Analysis of hint use serves many purposes and may be an obvious answer to *wheel-spinning*, where a student persists long past the point of productive effort [6]. It is also feasible to predict the problematic behaviors of hint misuse or hint abuse. Previous research has analyzed relationships between problem-related features (e.g., problem length, number of hints available, hint length) and student affect, behavior, and learning ([3][11][13][19]). Among other findings, hint length has been positively correlated with *gaming the system* [3], a behavior incorporating help abuse that is associated with poorer learning outcomes ([21][23]). Other research has indicated problems unrelated to the deliberate behavior of students. For example, poorly designed hints may lead to ineffective hint usage ([4][15]). Research also suggests that low-knowledge students, or those that need the most help, are the least likely to use it effectively ([2][3][18]).

In this paper, we present results from a randomized controlled trial (RCT) that examined how hint availability effected other aspects of student learning, including the time required for students to complete the assignment, presented using the ASSISTments online learning system [11]. To our surprise, we found that students who were given the option to request on-demand hints appeared to spend more time on tasks unrelated to the completion of the problem set (e.g., solve other problem sets, work on learning activities outside of ASSISTments, or engaged in activities external to the learning system). Specifically, these students took more time to complete the assignment even though they did not (a) spend significantly more time on task, (b) answer significantly more problems, or (c) make significantly more attempts per problem as compared to the control condition. The analyses presented herein explore this pattern more thoroughly, in order to contribute to the growing literature on help systems in online learning.

2. ASSISTMENTS
ASSISTments is an online learning system designed primarily for middle school mathematics. The platform allows teachers to easily create and assign their own problem sets (including questions, associated solutions, mistake messages, feedback) or to select from a set of *ASSISTments Certified Problems* (vetted by ASSISTment's expert team) ([11][22]). These problem sets simultaneously support student learning and serve as automated formative assessments that provide real-time data to teachers [11]. The platform is also used as a research tool to conduct RCTs

([8][16][26]). ASSISTments logs learning-related features at multiple granularities (e.g., problem text, problem type, student actions, timestamps, etc.). Figures 1 and 2 show screenshots of the types of ASSISTments problems used in the present work. Based on experimental condition, students were able to request hints, receive feedback messages, or simply answer the question.

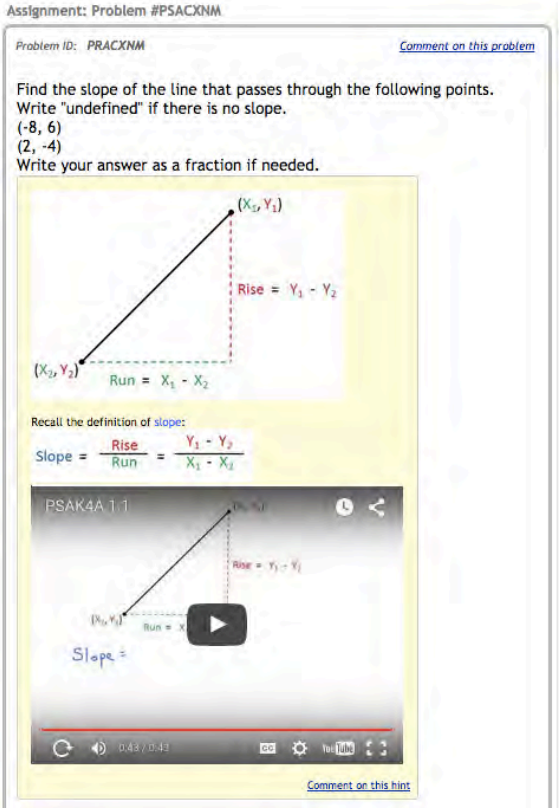


Figure 1. An example question from the hints-early condition, presented with its associated hints.

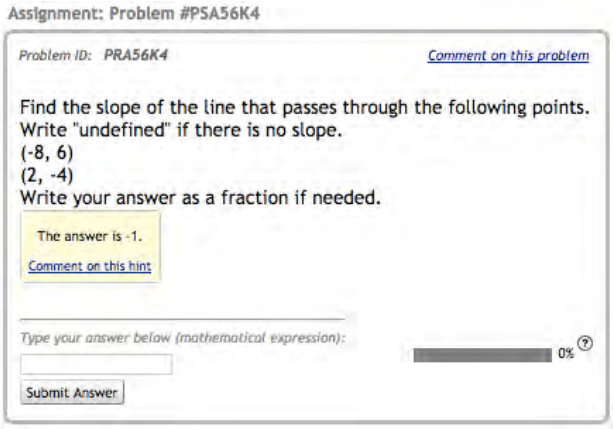


Figure 2. The same example question as presented in the no-hints-early condition.

3. METHODOLOGY

This study used an RCT design in which several linear presentations of a problem set were embedded within two conditions: a control condition with on-demand hints (*hints-early*, **HE**) or an experimental condition with on-demand hints only after the third problem (*no-hints-early*, **NHE**). The problem set for this study (available at [14]) was chosen from ASSISTments Certified content and was designed to address the 8th grade Common Core State Standard, "Finding Slope from Ordered Pairs," [17]. It was deployed within ASSISTments as a *Skill Builder*, a type of problem set requiring students to accurately answer three consecutive problems in order to complete the assignment.

Students were randomly assigned into one of 12 groups (6 control and 6 experimental) when they began the problem set. As depicted in Figure 3, students in each group saw the same 3 problems, but presentation order was randomized to minimize cheating (i.e., A-B-C, A-C-B, B-A-C, etc.). All students, regardless of condition, received immediate correctness feedback (e.g., "Sorry try again: '2' is not correct").

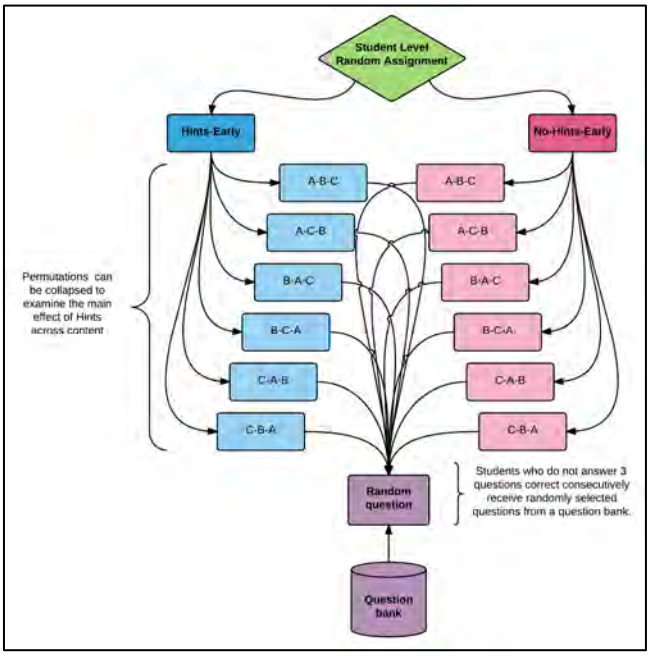


Figure 3. Research Design depicted as a flow chart.

In a Skill Builder, students are able to attempt each problem multiple times, but (in line with common practice) problem accuracy is calculated using binary correctness on the student's first attempt (1=Right, 0=Wrong) [10]. Students who did not answer the first three problems correctly were assigned additional problems randomly selected from a skill bank. In order to provide all students with adequate learning support, all students were permitted on-demand hints—regardless of condition—upon reaching these additional problems.

Figures 1 and 2 demonstrate how the interface differed by condition. In the HE condition, students could access hints at any time by clicking on a button in the lower right corner of their screen. The problem remained on the screen while video tutorials and text-based hints were simultaneously delivered (text-based hints ensured access when school firewalls or connectivity issues may have limited access to YouTube). In contrast, the NHE condition only offered a *Show Answer* button in the lower right corner of the screen during the first three problems (a design seen in early intelligent tutors [24]) allowing students who were stuck to move on to the next problem and eventually complete the assignment.

3.1 Student Populations

To help retain students' math skills, the Skill Builder in this study was one of many assigned as summer work at two suburban high

schools (henceforth Schools A and B) in the Northeastern U.S. **School A** was an agricultural/vocational high school that assigned this Skill Builder to 113 9th graders and 95 10th graders, along with numerous other Skill Builders (32 in 9th grade, 36 in 10th). **School B** was a high school without a known specialization; it assigned this Skill Builder (as well as 45 others) to 204 9th graders. Students worked on these assessments throughout the summer (Jun-Sept 2015) and data was harvested six months later.

Condition distributions were well matched for student gender (HE: 101 f., 86 m., 29 unknown vs. NHE: 93 f., 89 m., 14 unknown), school, grade level, and classroom section. Students in both conditions had the same **prior Skill Builder completion rate** (HE: *M*=0.91, *Mdn*=1.0; NHE: *M*=0.91, *Mdn*=1.0, *p*=.463), which was computed by dividing the sum of prior Skill Builders started by the number of prior Skill Builders completed (amongst all ASSISTments assignments experienced by students in the sample) . Analysis using Mann-Whitney U tests (which are robust to skew) with a Benjamini-Hochberg false-discovery rate post-hoc correction for multiple tests ($p<.05$) [7], yielded no significant differences between the two conditions on several measures including total number of problems solved, time per problem, and number of attempts.

3.2 Measures Considered

This study considered several measures pertaining to students' answers and hint patterns. As noted above, students only completed the Skill Builder when they correctly answered three consecutive questions using first attempts. However, students were able to attempt problems multiple times. Students wishing to advance to the next problem but unable to generate the correct answer were able to request a bottom-out hint. When hints were available, students had to view between 1 and 3 regular hints before they were able to obtain the bottom-out hint, which provided the correct answer. In the first three problems of NHE condition, students could select *Show Answer*, which displayed only the bottom-out hint, but no additional assistance.

Several measures based on these behavioral patterns were considered, including: **number of problems solved (PS), mean answer-attempts per problem (MAA)**, **total answer attempts (TAA)**, **total hint requests (THR)** and **mean hint requests per problem (MHR)**. Spanning conditions, participants required 9.12 problems on average (*Mdn*=9.0, *SD*=3.32) to complete the assignment. Spanning conditions and problems, students averaged 16.14 total answer attempts (*Mdn*=14.0, *SD*=10.72), or 1.72 answer attempts (*Mdn*=1.71, *SD*=0.78) per problem. On average, students requested approximately one hint per nine problems (*Mdn*=0.0 , *SD*=2.23) throughout the Skill Builder. There were no significant differences in the aforementioned measures by condition according to Mann-Whitney U tests conducted with false discovery rate post-hoc corrections.

Next, we assessed several time-based measures to determine how hints were affecting students' completion rates. Basic measures including the number of **days** and **weeks** it took for a student to finish the Skill Builder were considered. These measures were analyzed both by completion time and by week of completion. As the data was heavily skewed (most students finished in week 1), a Mann-Whitney U test was used to analyze completion time. Six months after beginning the study, when data was harvested, seventy-two students (18%) had not completed the Skill Builder. Students who completed the Skill Builder were grouped according to whether it had taken them *1*, *2*, *3*, or *4 or more* weeks to complete, while those who never finished the Skill Builder were labeled as *incomplete*. We also considered, **Completion time** (**CT**, in seconds), or the total time it took students to complete the assignment, which was calculated by subtracting the start time of the first problem from the end time of the last problem solved.

Because the time students spent solving a problem was skewed, with a median of 1.1 minutes (*M*=16.22 hr, *SD*=4.69 days, *Min*=2 sec, *Max*=74.96 days), this value was *winsorized* to 15 minutes (900 sec) in order to exclude irrelevant conditions (e.g., disconnection from the network, shifts between learning activities, off-task behavior). The fifteen-minute time frame accounted for 93% of the data.

The winsorized measures were used to calculate **time-on-problem (TOP,** in seconds**)** for each problem in the Skill Builder that the student attempted to solve (i.e., end time minus start time for each problem). This measure was subsequently used to generate several others, including **mean time-per-problem (MTPP),** which showed a mean of 2.62 min (*Mdn*=2.35 min, *SD*=1.78 min) across all students. For each student, **TOP** was also **totaled** across all attempted problems (**TOP-total**), resulting in a mean of 23.42 minutes (*Mdn*=20.72 min, *SD*=16.93 min) across all students. Finally, **total time-between-problems (TTBP)**, was calculated by subtracting TOP-total from each students' completion time. Readers should note that because students were allowed to return to this assignment over the course of the summer, these values were comparatively large (*M*=6.73 days, *Mdn*=43 sec, *SD*=14.49 days). However, as Table 1 shows, variation among students who took more than one week was minimal at the problem level.

Table 1. Mean values of time-based measures according to completion-time categories (weeks).

Week	PS	TOP-total	MTPP	TTBP	CT
1	9.15	20.2 m	2.2 m	0.48 d	0.49 d
2	10.04	35.9 m	3.7 m	10.1 d	10.1 d
3	9.00	38.2 m	4.4 m	18.5 d	18.5 d
≥ 4	11.81	38.6 m	3.3 m	40.8 d	40.8 d
Incomplete	5.55	16.9 m	3.6 m	5.4 d	N/A

Note. **PS – problems solved; TOP-total – total time on problem; MTPP – mean time per problem; TTBP – total time between problems; CT – completion time, m = minutes, d = days**

4. RESULTS

ASSISTments automatically logged data in analyzable form. The following subsections present the results on hint usage, problem attempts, skill builder completion, and time-on-problem.

4.1 Hint Usage and Problem Attempts

This study used four primary measures of student actions, including total answer attempts, mean answer attempts, total hint requests, and mean hint requests per problem. Because the two conditions in this study only applied to the first three problems (after which, students in the no-early-hints condition also had access to regular hints), we report on values for the first three problems and those that follow separately.

Table 2 presents significant differences both between and within-conditions. There were no significant differences between conditions with respect to the number of attempts per problem or the total number of attempts used in solving the first three problems of the Skill Builder. That is, the availability of hints in the first three problems did not effect the number of attempts used or the number of hints requested over the course of the experiment. Likewise, the significant differences observed within condition all trended in the same direction, suggesting little to no effect.

Table 2. Significant differences in answer attempts and hint requests by condition and within condition (p<.05).

Measure	HE vs. NHE		1st 3 vs. Other problems	
	1st 3	Others	HE	NHE
TAA	*NS*	*NS*	**Others > 1st3**	**Others > 1st3**
MAA	*NS*	*NS*	*NS*	**Others > 1st3**
THR	N/A	*NS*	**1st3 > Others**	N/A
MHR	N/A	*NS*	**1st3 > Others**	N/A

Note. **TAA – total answer attempts; MAA – mean answer attempts; THR – total hints requests; MHR – mean hint requests; HE – hints-early; NHE – no-hints-early; NS – not significant**

4.2 Hint Usage and Skill Builder Completion

One of the most important measures in this study was whether or not students were eventually able to demonstrate skill mastery by consecutively answering three of the Skill Builder questions accurately. Chi Squared tests revealed no significant difference between conditions in the proportion of students who did not complete the Skill Builder (X^2(1, N=412)=0.714, p=.398).

Non-completion in both conditions was associated with lower prior Skill Builder completion rates, suggesting that students' inability to master this Skill Builder was indicative of larger issues in completing their mathematics assignments (HE: U=1115.5, p<.001, NHE: U=471, p<.001). Non-completion was also associated with higher numbers of hint requests and answer attempts, both of which occurred across significantly fewer problems than worked by students who were able to complete the Skill Builder. Finally, non-completion was associated with significantly longer time worked across problems (**TOP-total**).

Despite nearly identical Skill Builder completion rates, the two conditions differed significantly in the time it took students to complete the problem set (HE: M=208.23 hrs, Mdn=38.55 min, NHE: M=67.52 hrs, Mdn=20.9 min, U=16835, p=.008). Specifically, as shown in Table 3, students in the no-hints-early condition completed the Skill Builder faster than those in the hints-early condition. These results were complemented by Chi Squared results that analyzed the distribution of students completing the assignment over several weeks, X^2(4, N=411)=8.981, p=.062. Again, this might seem obvious, as students who access hints tend to take longer to digest problem and feedback content, but further analysis suggests other factors should also be considered.

Table 3. Number of students per condition who completed the Skill Builder each week

Weeks	HE (N=215)	NHE (N=196)
1	125 (58%)	137 (70%)
2	15 (7%)	13 (7%)
3	5 (2%)	3 (1%)
≥ 4	30 (14%)	13 (7%)
Incomplete	40 (19%)	31 (16%)

Note. **HE – hints-early; NHE – no-hints-early**

4.3 Hint Usage and Time-on-Problem

Hint availability could effect time-on-problem (**TOP**) in more than one way, even when students use hints effectively. Students who need hints may be expected to answer more slowly than their peers, but powerful hints may actually reduce the time that a struggling student takes to complete a problem (compared to a situation in which the same student did not have access to hints).

Table 4 (calculated with the Benjamini-Hochberg correction) shows a complex interaction between time-per-problem and hint use, but overall there were few differences between conditions. On the whole, the use of (regular) hints lead to longer time on problem (**TOP**) measures, but the effect of bottom-out hints differed by condition. In both conditions, students who used bottom out hints took longer to complete problems than those who did not use them. However, those who used bottom-out hints in the HE condition took less time per problem than those who only requested one (regular) hint. The latter pattern could be indicative of *gaming* behavior, and this warrants further investigation, but it is also possible that students who quickly realized their mistakes clicked through to the bottom-out hint in order to start work on the next problem.

Results further indicated that differences were driven by hint use effects in the first three problems, where students who did not have access to hints (the NHE condition) were significantly slower at answering than those who did (HE) (M=1.92 min, Mdn=1.80 min vs M=1.65 min, Mdn=1.37 min). This was a predictable difference, as struggling students in the HE condition could ask for hints, thereby removing themselves from this calculation, while struggling students in the NHE condition could only remove themselves from this calculation by requesting a bottom-out hint.

Table 4. Time-on-problem comparison by condition (in minutes)

	Mean (SD)								Median			
	Regular Hints Requested						*Bottom-out Hint*		*Regular Hints Requested*			*Bottom-out Hint*
Condition	N	0 Hints	N	1 Hint	N	2 Hints	N		0 Hints	1 Hint	2 Hints	
First 3 Problems	373	1.78 (1.15)	103	3.62 (1.33)	0	N/A	167	2.98 (1.45)	1.48	3.85	N/A	2.95
HE	191	1.65* (1.17)	103	3.62 (1.33)	0	N/A	81	3.47* (1.33)	1.37*	3.85	N/A	3.43*
NHE	182	1.92* (1.13)	0	N/A	0	N/A	86	2.55* (1.43)	1.80*	N/A	N/A	2.53*
Other Problems	366	1.52 (0.92)	22	2.52 (1.93)	59	3.27 (1.23)	56	3.27 (1.23)	1.33	1.78	3.02	2.98
HE	190	1.50 (0.87)	13	2.02 (1.92)	30	3.20 (1.33)	29	3.23 (1.33)	1.33	1.65	2.92	2.93
NHE	176	1.53 (0.98)	9	3.25 (1.82)	29	3.37 (1.15)	27	3.28 (1.15)	1.42	3.65	3.47	3.02
All Problems	377	1.58 (0.78)	113	3.50 (1.37)	58	3.32 (1.17)	174	3.05 (1.35)	1.53	3.65	3.22	3.03
HE	195	1.50 (0.73)	104	3.53 (1.33)	29	3.28 (1.20)	87	3.45* (1.27)	1.52	3.63	2.93	3.40*
NHE	182	1.67 (0.83)	9	3.25 (1.82)	29	3.37 (1.15)	87	2.65* (1.33)	1.57	3.65	3.47	2.70

Note. **Units are in minutes. *p<.05. N – number of students; HE – hints-early; NHE – no-hints-early.**

Significant differences within and between conditions (summarized in Table 5) showed trends that suggested that behavior in the first three problems was driving the differences between the two conditions, where hint-access was restricted to the students in the HE condition. Interestingly, in the first three problems the mean time per problem was statistically similar. That is, for the first three problems, the HE and NHE condition did not differ overall, which suggests the need for understanding individual differences, such as those highlighted in Table 4. The significant differences between conditions emerged primarily in total time between problems (**TTBP**) and in the total completion time (**CT**), with students in the hints-early condition showing larger values for both measures.

Table 5. Time Measures per Condition (p<.05).

	HE vs. NHE		1st 3 vs. Other problems	
	1st 3	Others	HE	NHE
MTPP	NS	NS	**1st3 > Others**	NS
TTBP	**HE > NHE**	NS	NS	**Others > 1st3**
CT	**HE > NHE**	NS	NS	**Others > 1st3**

Note. **MTPP – mean time-per-problem; TTBP – total time between problems; CT – completion time; HE – hints-early; NHE – no-hints-early; NS – not significant**

Further analyses revealed complementary patterns in within-condition differences. Students in the hints-early condition had significantly higher mean time-per-problem (MTPP) on the first three problems than they did on later problems (M=3.67 min, Mdn=2.63 min vs. M=2.17 min, Mdn=1.98 min, U=13281, p<.001), suggesting that those who effectively used these hints in the first three problems were learning the material well enough to complete later problems more efficiently. There were no significant differences in this group for other time-based measures (**TTBP** or **CT**). In contrast, students in the no-hints-early condition showed no significant differences for **MTPP**, but had longer **TTBP** and **CT** patterns for later problems than for the first three problems.

5. DISCUSSION

The present experiment was designed to explore the effects of ASSISTments' on-demand hints system. For ethical reasons, we limited differences between the control condition (providing hints) and the experimental condition (withholding hints) to the first three problems. All students had access to hints following the third problem to retain overall learning. However, effects could be seen even after students had moved past these first three problems.

The data used in the study was collected from one of many Skill Builders assigned to students for summer work. We explored the data using several different measures, extracting information about the number of attempts each student made, the number of hints (regular or bottom-out) they requested, and the length of time needed to complete the assignment.

Some findings were quite predictable, as reading hints would take more time than simply answering problems, assuming students were assigned problems that matched their current ability. However, other findings were more surprising. Even though students made the same number of attempts per problem and per assignment, those in the HE condition took significantly longer to complete the Skill Builder.

Students in the HE condition also spent relatively more time between problems compared to those in the no-hints-early condition, but only during the first three problems, where conditions were truly distinct. One interpretation of this finding is that students in the HE condition were taking more time between problems to process the new material they were learning. An alternative explanation is that students were procrastinating—deliberately putting off working on the Skill Builder out of difficulty or apathy (as summer work is highly self-regulated). These students could have been seeking out an easier Skill Builder to work on or may have spent their time doing something completely unrelated. Still, this latter interpretation may not be detrimental if students were using the time to work on other assignments. As Baker and colleagues have suggested [5], a student that goes off task and is able to re-engage afterwards may be more productive in the long run than those who persist at all costs.

6. CONCLUSION

This work presented an investigation of how students completing summer work responded to having or not having hints available on the first three problems of a Skill Builder assignment within the ASSISTments online learning system. When hints were available from the start, students were more likely to delay work in comparison to students for whom step-wise hints were only available after the third problem. When hints were always available, participants took significantly more time to complete the Skill Builder. We interpreted the difference in completion times as an opportunity to re-engage towards more productive math learning. In future work, we plan to conduct a similar study during the school year to examine how results differ in a more controlled and less self-regulated learning environment.

7. ACKNOWLEDGEMENTS

We gratefully acknowledge funding from the NSF (ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428 & DRL-1031398), the US Dept. of Ed (IES R305A120125 & R305C100024 and GAANN), and the ONR.

8. REFERENCES

[1] Aleven, V., McLaren, B., Roll, I., & Koedinger, K. 2006. Toward Meta-Cognitive Tutoring: A Model of Help-Seeking with a Cognitive Tutor. *Int J Artif Int in Ed*, *16*, 101-130.

[2] Aleven, V., Stahl, E., Schworm, S., Fischer, F., Wallace, R. 2003. Help seeking and help design in interactive learning environments. *Rev Educ Res*, *73*(3), 277-320.

[3] Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. 2004. Off-Task Behavior in the Cognitive Tutor Classroom: When Students Game The System. *Proc ACM CHI*, 383-390.

[4] Baker, R.S.J.d., de Carvalho, A.M.J.A., Raspat, J., Aleven, V., Corbett, A.T., Koedinger, K.R. 2009. Educational Software Features that Encourage and Discourage "Gaming the System". *Proc 14th Int Conf Artif Int in Ed*, 475-482.

[5] Baker, R.S.J.d., Moore, G., Wagner, A., Kalka, J., Karabinos, M., Ashe, C., Yaron, D. 2011. The Dynamics Between Student Affect and Behavior Occuring Outside of Educational Software. *Proceedings of the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction*.

[6] Beck, J., & Gong, Y. 2013. Wheel-spinning: Students who fail to master a skill. *Arti Int in Ed*. Berlin: Springer.

[7] Benjamini, Y. and Hochberg, Y. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society, 289-300.

[8] Broderick, Z., O'Connor, C., Mulcahy, C., Heffernan, N. & Heffernan, C. 2011. Increasing Parent Engagement in Student Learning Using an Intelligent Tutoring System. *J Int Learn Res,* 22(4):523-550.

[9] Campuzano, L., Dynarski, M., Agodini, R., & Rall, K. 2009. Effectiveness of reading and mathematics software products: Findings from two student cohorts. Washington, DC: U.S. Dept Ed, Inst Ed Sci.

[10] Corbett, A. T., & Anderson, J. R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Model User-Adap Intrc*, *4*(4), 253-278.

[11] Heffernan, N., & Heffernan, C. 2014. The ASSISTments Ecosystem: Building a Platform that Brings Scientists and Teachers Together for Minimally Invasive Research on Human Learning & Teaching. *Int J AIED* 24(4),470-97.

[12] Heiner, C., Beck, J., & Mostow, J. 2004. Improving the help selection policy in a Reading Tutor that listens. *InSTILL/ ICALL Symposium.*

[13] Inventado, P.S. & Scupelli, P. 2015. Data-Driven Design Pattern Production: A Case Study on the ASSISTments Online Learning System. *Proc 20th Euro Conf Pattern Languages of Programs*.

[14] Inventado, P.S., Scupelli, P., Van Inwegen, E.G., Ostrow, K.S., Heffernan, N., Baker, R.S., Slater, S., & Ocumpaugh, J. 2015. Materials for *Hint Availability Slows Completion Times in Summer Work.* Retrieved from https://goo.gl/xyli5h

[15] Koedinger, K.R., & Aleven, V. 2007. Exploring the assistance dilemma in experiments with Cognitive Tutors. *Educ Pscyhol Rev* 19.3: 239-264.

[16] Li, S., Xiong, X., & Beck, J. 2013. Modeling student retention in an environment with delayed testing. *Int Educ Data Mining Society*, 328-329

[17] Natl Gov Assoc Ctr Best Practices, Council of Chief State School Officers. 2010. Common Core State Stds. Washington D.C.

[18] Nelson-Le Gall, S. 1987. Necessary and unnecessary help-seeking in children. *J Genetic Pscyhol,* 148, 53–62.

[19] Ocumpaugh, J., Baker, R.S, Rodrigo, M.M.T. 2015. *Baker Rodrigo Ocumpaugh Monitoring Protocol (BROMP) 2.0 Technical & Training Manual.* NY, NY: Teachers College, Columbia U. Manila, Philippines: Ateneo Laboratory for the Learn Sciences.

[20] Pane, J.F., McCaffrey, D.F., Slaughter, M.E., Steele, J.L., & Ikemoto, G.S. 2010. An experiment to evaluate the efficacy of Cognitive Tutor geometry. *J Res Educ Eff*, 3(3), 254–281.

[21] Pardos, Z., Baker, R.S., San Pedro, M.O.C.Z., Gowda, S., Gowda, S. 2014. Affective states & state tests: Investigating how affect and engagement during the school year predict end of year learning outcomes. *J Learn Anlytc*, 1(1), 107-28.

[22] Razzaq, L.M., & Heffernan, N.T. 2009. To Tutor or Not to Tutor: That is the Question. *AIED,* 457-464.

[23] San Pedro, M.O., Baker, R., Heffernan, N., Ocumpaugh, J. 2015. Exploring College Major Choice and Middle School Student Behavior, Affect and Learning: What Happens to Students Who Game the System*? Proc 5th Int Learn Anlytc Know*, 36-40.

[24] Schofield, J. W. 1995. Computers and Classroom Culture. *Cambridge University Press*.

[25] VanLehn, K. 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educ Psychol,* 46(4), 197-221.

[26] Whorton, S. 2013. Can a computer adaptive assessment system determine, better than traditional methods, whether students know mathematics skills? MA thesis, Computer Science Department, Worcester Polytechnic Institute.

[27] Wijekumar, K., Meyer, B., & Lei, P. 2012. Large-scale RCT with 4th graders using intelligent tutoring of the structure strategy to improve nonfiction reading comprehension. *Educ Tech Res Dev*, 60(6), 987-1013.

On Competition for Undergraduate Co-op Placements: A Graph Mining Approach

Yuheng Jiang and Lukasz Golab
University of Waterloo, Canada
{y29jiang,lgolab}@uwaterloo.ca

ABSTRACT

We propose a graph mining methodology to analyze the relationships among academic programs from the point of view of co-operative education. The input consists of student - job interview pairs, with each student labelled with his or her academic program. From this input, we build a weighted directed graph, which we refer to as a program graph, in which vertices correspond to academic programs and edge weights denote the percentage of jobs that interviewed at least one student from both programs. We show that various properties of this graph have natural interpretations in terms of the relationships among academic programs and competition for co-op jobs. We also present a case study that illustrates the utility of the proposed methodology.

1. INTRODUCTION

According to the World Association for Cooperative and Work-integrated Education, 275 institutions from 37 countries have implemented cooperative education (co-op) programs [17]. Co-op experiences are vital because they supplement students' classroom skills and help them to gain practical experience.

We propose a graph mining methodology to analyze the relationships and competition among academic programs in the context of co-op. Our motivation is threefold. First, with academic institutions introducing new programs in recent years [6, 15], it is often unclear how one program differs from another. As a result, employers may not know which programs to advertise their jobs to and students may not realize that they qualify for a job targeted to a related program (e.g., Computer Science vs. Software Engineering). Understanding similarities among programs can lead to more effective job and academic classification schemes and therefore can help match job opportunities with qualified students. This analysis can also help students choose programs of study that correspond to their desired careers. Second, data from the co-op system may be used to identify multi-disciplinary programs that enable their students to obtain various types of jobs. This issue is becoming increasingly important given the recent rise in popularity of multi-disciplinary and well-rounded education [1, 2, 5, 10, 16]. Third, analyzing co-op job data can reveal jobs that are exclusive to particular departments, and, conversely, departments whose students compete for jobs with students from other departments. The university can choose to attract more employers that offer jobs to programs facing strong competition. Thus, the problems we study in this paper are critical to co-operative education from the student's, employer's and institution's perspective.

While some of these questions have been raised in prior work (details in Section 2), we propose a data-driven technique for answering them. Our input consists of student - job interview pairs, with each student labelled with his or her academic program. We transform this input to a graph, which we refer to as a *program graph*, in which vertices correspond to academic programs and edge weights denote the percentage of jobs that interviewed at least one student from both programs. Thus, the larger the edge weight, the stronger the relationship and competition between two programs.

Within the program graph, we are interested in vertices forming clusters or communities, vertices that are connected to many such clusters, and vertices that are strongly connected to their neighbours. As we will show, these graph properties have natural interpretations in the context of co-op. Graph clustering and community detection determine groups of related programs whose students interview for the same types of jobs; programs with connections to multiple clusters are likely to be multi-disciplinary; and programs with strong connections to their immediate neighbours face strong competition for jobs.

2. RELATED WORK

The majority of related work qualitatively or statistically analyzed co-op education through survey data with fewer than 100 entries. To the best of our knowledge, the first research work that used a large-scale data-driven methodology was our previous work [9]. We analyzed satisfaction with the co-op process using three years of evaluation data (students' evaluations of their employers and employers' evaluations of students). We found that students received better evaluations in their senior years, but they rated their first employer the highest. We also found that senior students outperformed junior students in work placements abroad, and extended work terms at the same employer (spanning more than one academic term) did not increase student satisfaction. In this paper, we target a different problem of understanding the relationships among academic programs.

In the context of academic programs, Wilson and other researchers urged traditional academic disciplines to be updated to better reflect reality [6, 15]. Furthermore, Hesketh found that employers have trouble advertising to specific programs and instead they ad-

vertise based on desired skillsets [8]. As we will show, clusters in the program graph indicate similar programs and suggest related programs that employers can advertise their jobs to. Additionally, it was suggested that programs can be evaluated based on their students' ability to obtain jobs [7, 14], which is a question that can be answered with the help of our methodology. Also, while the importance of multi-disciplinary education has been widely recognized [1, 2, 5, 10, 16], we propose a data-driven methodology for analyzing whether students from a particular academic program qualify for different types of jobs.

3. METHODOLOGY

We are given a dataset corresponding to student - job interview pairs, with each student labeled with his or her academic program and each interview associated with a job ID. We propose a methodology that relies on transforming the student-job interview pairs to an edge-weighted directed graph $G = (V, E)$, with a set of vertices V and a set of edges E. Vertices correspond to academic programs and edges represent relationships among programs. Let e_{ij} be the weight of the edge E_{ij} from vertex v_i to v_j, and let J_i be the list of distinct jobs that interviewed students from program v_i. We define e_{ij} as the fraction of jobs that interviewed at least one student from both programs; i.e., the fraction of jobs in J_i that also appear in J_j:

$$e_{ij} = \frac{|J_i \cap J_j|}{|J_i|} \quad (1)$$

This can also be interpreted as a conditional probability that a job interviewed at least one student from program v_j given that it interviewed at least one student from program v_i.

The direction of edges is important. For a program node v_i, an incoming edge weight from v_j measures the fraction of jobs in J_j that also interviewed at least one student from v_i. Thus, a large incoming edge weight of v_i from v_j means that most jobs interviewing at least one student from v_j also interviewed at least one student from v_i. Conversely, a large outgoing edge weight from v_i to v_j means that most jobs interviewing at least one student from v_i also interviewed at least one student from the other program.

We give an example in Table 1, which corresponds to 4 jobs, 9 interviews and 8 students from three programs (A, B and C). The job lists for each program are: $J_A = \{1, 2, 3\}$, $J_B = \{1, 2\}$, and $J_C = \{2, 4\}$. The corresponding program graph is shown in Figure 1, and the edges are colour-coded by the source vertex. The edge weight from Program A to Program B is $|\{1, 2\}|/|\{1, 2, 3\}| = 2/3 = 0.67$, meaning that 67 percent of jobs that interviewed at least one student from Program A also interviewed at least one student from Program B. The edge weight from Program B to Program A is $|\{1, 2\}|/|\{1, 2\}| = 2/2 = 1$, meaning that every job which interviewed a student from program B also interviewed a student from program A. Thus, the larger the edge weight, the stronger the relationship and competition between two programs.

Our definition of edge weights assumes that a relationship between two programs exists if at least one student from both programs *interviewed* for the same job; if there are many such jobs, then the edge weight will be larger.

Having explained how the program graph is constructed, we now clarify how properties of the program graph are related to the types and extent of relationships among academic programs in the context of co-op jobs:

Table 1: Sample interview data

Student ID	Program Name	Job ID
1	A	1
2	C	2
3	B	1
3	B	2
4	B	1
5	A	2
6	A	3
7	C	2
8	C	4

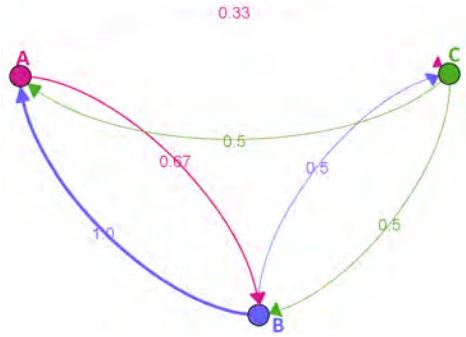


Figure 1: An example of a program graph

- **Clusters:** Clusters in a graph represent closely connected vertices. In our context, clusters represent related programs whose students interview for (mostly) the same jobs.
- **Outliers:** Given a graph clustering, we define outliers as vertices that have strong connections to other vertices from multiple clusters (as opposed to "normal" vertices connected mostly to other vertices within the same cluster). In our analysis, outliers correspond to multi-disciplinary programs: students from those programs have interviews in common with students from several different program clusters.
- **Fan-out:** (Weighted) fan-out measures the (weighted) number of outgoing edges of a vertex. In our context, weighted fan-out corresponds to the competition that a program faces from other programs. High weighted fan-out means that most jobs interviewing at least one student from the given program also interviewed students from other programs. As we will explain shortly, we use a modified version of standard weighted fan-out that takes into account the fact that our edge weights are defined in terms of set intersections (of the job sets of different programs).

In the remainder of this section, we describe the graph algorithms that may be used to identify program clusters, multi-disciplinary programs and programs facing strong competition.

3.1 Finding Clusters of Similar Programs

We use two techniques to find clusters of similar programs: near-clique finding and community detection.

The density of a graph (or subgraph) is the number of edges divided by the maximum possible number of edges, i.e., $\frac{|E|}{|V|*(|V|-1)}$. A clique is a group of vertices that are fully connected and therefore have a density of one. A near-clique is a group of vertices where

the subgraph consisting of them and their edges has a density of nearly one, i.e., a group of vertices that is nearly fully connected. However, since our program graph is weighted and directed, we want to find near-cliques with large edge weights. To do this, we first remove all edges from the program graph except the five percent with the largest edge weights. The resulting graph may leave some vertices disconnected, while other pairs of vertices may only have an incoming or an outgoing edge. Then, we remove edge directions and simply retain an edge between two programs if there is either an incoming or an outgoing edge. Finally, we return all near-cliques from the resulting graph with density of at least 0.8.

In addition to identifying densely connected subgraphs via near-clique finding, we use the Louvain Modularity algorithm [4] to partition the vertices into disjoint clusters (communities), such that vertices with the same cluster are densely connected and vertices in different clusters are sparsely connected. This algorithm is included in many graph mining tools such as Gephi [3] and aims to maximize *modularity*, which compares the sum of the weights of intra-cluster edges resulting from given clustering with that of a randomly connected graph with the same number of edges [13].

Newman [12] introduced modularity for weighted undirected graphs. We translate this metric to weighted directed graphs as follows. Let c_i be the community that a vertex v_i belongs to, and $m = \sum_{ij} e_{ij}$, i.e., the sum of all the edge weights in the graph. The fraction of the edge weights that are intra-cluster is $\frac{1}{m}\sum_j e_{ij}\delta(c_i, c_j)$, where $\delta(c_i, c_j)$ is equal to 1 if $c_i = c_j$ (i.e. vertices v_i and v_j belong to the same cluster) and 0 otherwise.

Let $k_i = \sum_j e_{ij}$ (i.e., the sum of the weights of the edges that connect to vertex v_i). Consider another graph in which the fan-outs of all the vertices are the same but the edges are randomly connected. In such a graph, the probability of an edge existing between vertices v_i and v_j is $\frac{k_i k_j}{2m}$. The modularity of a graph clustering is defined as:

$$Q = \frac{1}{m}\sum_{i,j}(e_{ij} - \frac{k_i k_j}{m})\delta(c_i, c_j) \quad (2)$$

$Q = 0$ means that the community detection result is no better than random. The maximum value for Q is 1. Higher modularity indicates more effective partitioning with more intra-cluster edges and fewer inter-cluster edges.

The Louvain Modularity method is iterative and includes two phases. In the first phase, each vertex starts in a different community. Then, for each vertex v_i, we compute the gain in modularity if v_i is moved to the community that its neighbour (v_j) belongs to. If the gain is positive, the change happens; otherwise v_i remains in its original community. This process is repeated iteratively and sequentially until no further improvements can be made. The outcome of the first phase is only a local optimum of modularity since the order of processing of the vertices will affect the result. In the second phase, a new graph is created such that the vertices are the communities obtained in the first phase, and edge weights are the sums of edge weights between vertices in the two communities. We reapply the process in the first phase on this new graph. The algorithm stops when maximum modularity is reached. To account for the effect of order, we run this algorithm multiple times and keep the result with the highest modularity.

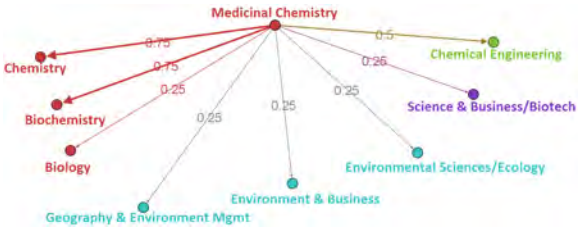


Figure 2: Direct competitors of Medicinal Chemistry, colour-coded by clusters

One characteristic of this algorithm is that it avoids creating small clusters. Lambiotte et al. [11] add a *resolution parameter* t to control the number of clusters. The new modularity definition is shown in Equation 3. The default t value is 1; smaller values of t lead to more and smaller communities.

$$Q_{new}(t) = (1-t) + \frac{1}{m}\sum_{i,j}(e_{ij}t - \frac{k_i k_j}{m})\delta(c_i, c_j) \quad (3)$$

3.2 Finding Multi-Disciplinary Programs

To find multi-disciplinary programs, we start with the clusters/communities obtained by the Louvain Modularity algorithm. Intuitively, if an academic program has strong connections to other programs from multiple clusters (each of which corresponds to different types of jobs), it may be multi-disciplinary.

For each program, we propose a multi-disciplinary score as follows. For each cluster c_i identified by the Louvain Modularity algorithm, let p_i be the fraction of the total weight of the outgoing edges from the given program to the programs only in c_i. Then, for a given program, we compute the entropy of the distribution of edge weights among different communities simply as $\sum_i -p_i \log_2 p_i$. High entropy means that the given program has strong links to programs in multiple clusters and therefore may be multi-disciplinary.

We illustrate this concept with an example. Suppose that students in the Medicinal Chemistry program had interviews in common with students from eight other programs belonging to four clusters, labeled red, blue, purple, and green, as shown in Figure 2, with vertices colour-coded by their clusters. Only the outgoing edges from Medicinal Chemistry are relevant since they represent the percentage of jobs from $J_{MedicinalChemistry}$ that also interviews students from its neighbour programs. The sum of all out-going edge weights of Medicinal Chemistry is 3.25. $p_{red} = (\sum_{i \in \text{red cluster}} e_{MedicinalChemistry,i})/3.25 = (0.75 + 0.75 + 0.25)/3.25 = 0.54$, which is the sum of weights of edges from Medicinal Chemistry to the programs in the red cluster. Similarly, $p_{blue} = 0.23$, $p_{green} = 0.15$, and $p_{purple} = 0.08$. Thus, the multidisciplinary score of Medicinal Chemistry is $-p_{red}\log_2 p_{red} - p_{blue}\log_2 p_{blue} - p_{purple}\log_2 p_{purple} - p_{green}\log_2 p_{green} = 1.67$.

3.3 Finding Programs Facing Competition

We define the extent of competition that a program faces using a "set fan-out" metric. We want to compute the fraction of jobs that interviewed students from the given program which also interviewed at least one student from another program. For a given vertex (program) v_i, we define:

$$\text{Set Fan Out}_i = \frac{|\cup_{j \neq i} (J_i \cap J_j)|}{|J_i|} \quad (4)$$

A set fan-out of zero means that all the jobs that interviewed at least one student from program v_i only interviewed students from v_i and no other program. Students from such a program may have specialized skills that students from other programs do not have. A set fan-out of one means that every job that interviewed at least one student from program v_i also interviewed at least one student from another program. In other words, there were no jobs that exclusively interviewed students from v_i and therefore students from v_i may be facing strong competition for jobs.

Returning to Table 1, $J_A = \{1, 2, 3\}$, $J_B = \{1, 2\}$, and $J_C = \{2, 4\}$. For Program A, its set fan-out is $\frac{|(J_A \cap J_B) \cup (J_A \cap J_C)|}{|J_A|} = \frac{|\{1,2\}|}{|\{1,2,3\}|} = \frac{2}{3} = 0.67$. It means that students from Program A competed with students from other programs in 67 percent of their jobs. 33 percent of jobs that interviewed students from Program A did not interview students from other programs. The set fan-out for Program B is 1 and for Program C it is 0.5.

4. CASE STUDY

We now describe a case study that illustrates the utility of the proposed methodology. To carry out the analysis, we used the Gephi toolkit [3] which includes the Louvain Modularity algorithm. We used data from a large Canadian university including all interviews taking place in summer 2014, for co-op jobs taking place in Fall 2014. For each student - interview pair, the dataset includes the student's academic program and year, and job information such as the company name, job title, and targeted programs and academic years. The dataset consists of 4,194 students from 93 academic programs, 2,890 jobs and 16,855 interviews. On average, each job interviewed 5.8 students and each student had 4 interviews.

This academic institution has six faculties, each comprised of a number of academic programs: Science (programs include Physics and Earth Sciences), Mathematics (programs include Computer Science and Actuarial Science), Engineering (programs include Electrical, Mechanical, Civil, etc.), Arts (programs include Economics, Psychology and Sociology), Environment (programs include Planning and Geomatics) and Applied Health Science (AHS) (programs include Kinesiology and Recreation and Leisure Studies). All Engineering programs and several programs from other faculties (mainly Mathematics) have mandatory co-op education; other programs have optional co-op. As a result, most of the students and jobs in our dataset are from Engineering and Mathematics.

Rather than using all available data, we build the program graph using only the interviews of *senior* students (in their third and fourth academic years). Junior-level jobs tend to be less specialized, meaning that (junior) students from many different departments may qualify for an interview. In particular, we noticed that entry-level computer programming jobs interview students from many programs, including those outside computing. By focusing on senior students, we avoid generating edges in the program graph that correspond to junior-level jobs and may not truly indicate a relationship between programs. The resulting program graph contains 88 vertices (corresponding to programs that have at least two senior students in co-op) and 1,315 pairs of directed edges.

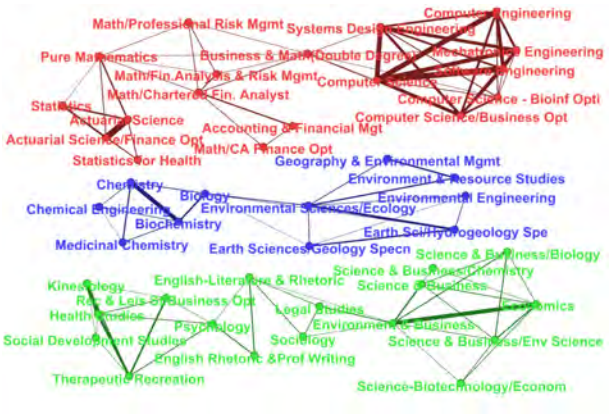


Figure 3: Vertices and edges participating in near-cliques

The program graph is a single connected component, i.e., there exists a path from every vertex to another. Its density is 0.34, meaning that one third of all possible program pairs had at least one interview in common. On average, the length of the shortest path between any two vertices is 1.7 and the diameter of the graph (i.e., the maximum length of any shortest path between two vertices) is three. The number of edges per vertex ranges from 4 to 66, with an average of 30.

4.1 Finding Clusters of Similar Programs

4.1.1 Near-Clique Finding

We begin by identifying near-cliques in the program graph (but considering only the five percent of edges with the largest weights, as described in Section 3). Figure 3 plots a subgraph of the program graph containing only the 46 vertices and 104 edges (in the top 5 percent of edge weights) that participate in the 25 near-cliques that we found. Three groups of programs appear to participate in the near-cliques, and we use a different colour for each. The larger the edge weight, the thicker the edge.

The red group at the top contains programs related to computing and maths. There is one near-clique with Software Engineering, Computer Engineering, Computer Science, Systems Design Engineering and Mechatronics Engineering. This suggests that Systems Design and Mechatronics students compete (interview) for software and programming jobs with students from core computing programs such as Computer Science. There are also two smaller near-cliques corresponding to Statistics/Actuarial Science and Accounting/Financial Analysis. Additionally, Pure Mathematics is connected to both of these; in fact Pure Mathematics students had interviews in common with students from 18 other programs. This suggests that Pure Mathematics students also interview for jobs in statistics, finance and business. Upon further inspection, we found that most such jobs were in financial trading.

The blue group of vertices in the middle includes two near-cliques: one with Chemistry-related programs and one with Earth Science and Environment-related programs. Based on these observations, the university may choose to either merge some of these related programs or redesign them to remove some of the overlap.

The green group at the bottom shows interesting connections. For instance, Economics seems strongly connected to Science & Business and Environment & Business, suggesting that these joint pro-

grams focus more on business than science (otherwise they would be connected with programs such as Chemistry and Environmental Engineering). Furthermore, there is a near-clique with seemingly unrelated programs: Sociology, Legal Studies, English-Literature & Rhetoric and Environment & Business; the first three are in the faculty of Arts while the last one is in the faculty of Environment. Upon further inspection, we found that the jobs these programs competed for were mainly in marketing and communications.

4.1.2 Community Detection

Next, we run the Louvain Modularity algorithm with different values of the resolution parameter to obtain a partitioning of the vertices into different numbers of communities, from 2 to 7. For example, Figure 4 shows the 7 communities we found, with each community in a different colour. For readability, we only include the edges in the top 5 percent of largest weights. Notice that Figure 3 is a subgraph of Figure 4, so all the near-cliques identified there are also visible here.

With these seven clusters, we obtain a partitioning into Engineering/Computing, Math/Finance, Natural Sciences, Social Sciences, Science &Business, Environment, and Health Sciences. Note that some engineering programs such as Chemical are placed in the Natural Sciences cluster and others such as Civil and Geological are placed in the Environment cluster. With only four clusters (illustration omitted for brevity), we obtain Engineering/Computing, Math/Finance, Natural Science/Environment, and Social/Health Science. With only two clusters (illustration omitted for brevity), we distinguish between Math/Engineering and Natural/Social Science programs.

4.2 Finding Multi-Disciplinary Programs

Recall that our methodology for identifying multi-disciplinary programs requires a clustering; then, for each program, we compute the entropy of its edge weight distribution across different clusters. We use the seven clusters from Figure 4 and obtained entropy values between 0.64 and 1.89. The top five multi-disciplinary programs (highest entropy) are: Science & Business/Biochemistry, English Literary Studies, Science & Business/Environmental Science, Biology, and Science & Business. The top five least multi-disciplinary programs are: Geological Engineering, Software Engineering, French, Mechatronics Engineering and Civil Engineering. Not surprisingly, joint programs of the form Science & Business were identified as multi-disciplinary while specialized engineering programs were not.

4.3 Finding Programs Facing Competition

We now search for programs with high set fan-out, i.e., those with few jobs that interviewed students only from that particular program. We found that for about half the programs, over 90 percent of the jobs that interviewed a student from a particular program also interviewed at least one student from another program. Thus, competition for jobs among academic programs appears relatively high. In particular, 16 programs, including Business & Mathematics, did not have any jobs that interviewed only their students (the jobs for which these students interviewed were computing-related or financial). Most of these 16 programs were small (only 3-4 senior co-op students). There were few jobs that specifically target these programs, so students from these programs had to interview for jobs advertised to other programs.

On the other hand, there were 8 programs where more than 30 percent of the jobs that interviewed at least one of their students did not

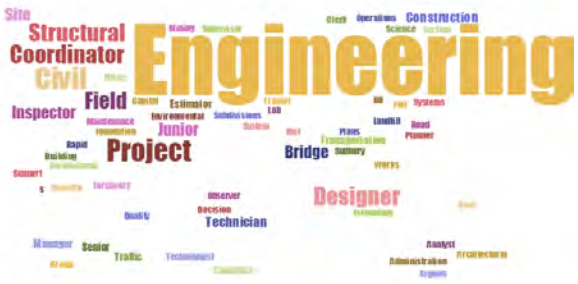


Figure 5: Word cloud of job titles of 70 Civil Engineering jobs that only interviewed students from Civil Engineering

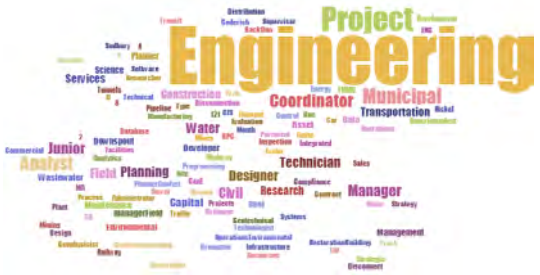


Figure 6: Word cloud of job titles of 85 Civil Engineering jobs that also interviewed students from other programs

interview students from any other program. They are Mathematical Studies/Business, Environmental Science - Geoscience, Information Technology Management, Accounting & Financial Management, Kinesiology, Chemical Engineering, Mechanical Engineering, and Civil Engineering. Upon inspection of the 70 jobs that interviewed only Civil Engineering students, we found that the job titles reflected expertise that is specific to this program, such as "structural", "field inspector", "bridge", "traffic" and "transportation" (see the word cloud in Figure 5). However, the remaining 85 jobs that interviewed Civil Engineering students also interviewed students from other programs, mostly other engineering programs such as Environmental, Mechanical and Geological Engineering. We show a word cloud of these job titles in Figure 6; notice that it includes more general keywords as compared to those in Figure 5. Thus, it appears that there may not be enough specialized jobs for programs such as Civil Engineering and some students within such programs compete for a broader set of jobs.

5. CONCLUSIONS

We presented a data-driven solution towards improving the co-operative education process. We observed that academic programs are typically used by students and employers to advertise and search for jobs, but it is not always clear how one program differs from another, especially given that universities have recently been creating new programs. In response to this problem, we developed a methodology to characterize the relationships among academic programs with respect to the job interviews obtained by students from these programs. The insight behind the methodology was to transform co-op interview data into a *program graph*, which revealed that students from certain programs interview for the same jobs as those from other programs. We proposed graph analyses such as finding communities, finding vertices connected to many communities, and finding vertices strongly connected to their neighbours to describe the program relationships.

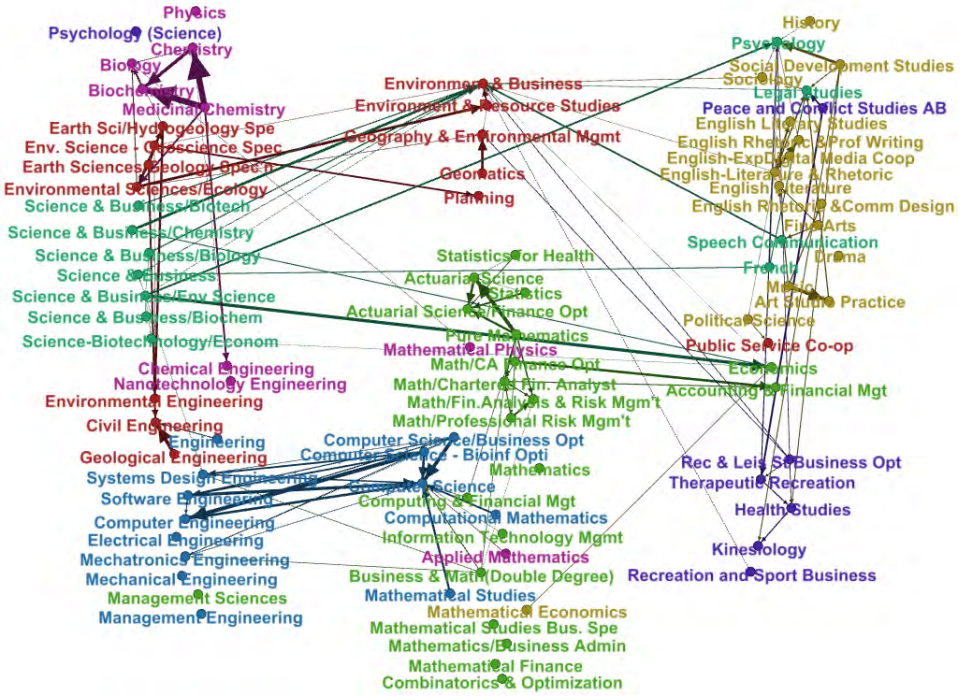


Figure 4: Clustering of the program graph into seven communities

We applied the proposed methodology on a large co-op data set from a major Canadian university. Our findings and their significance may be summarized as follows.

The clustering and community detection results (Section 4.1) correspond to job categories and academic specializations, which are not always evident from the University's academic structure. This suggests a job classification hierarchy to help advertise jobs to groups of related programs. Our results can also help students plan their academic and employment careers.

In Section 4.2, we identified multi-disciplinary programs which have strong connections to multiple clusters. These results can help students select programs that will give them broad skills and job qualifications, and can help institutions confirm that programs designed to be multi-disciplinary are producing students who qualify (i.e., are able to obtain interviews) for various types of jobs.

In Section 4.3, we identified programs where there were no jobs that only interviewed students from that particular program. That is, students from that program always competed for jobs with students from other programs. The university may wish to attract more employers that offer jobs to these under-represented programs.

6. REFERENCES

[1] R. Barnett. Supercomplexity and the curriculum. *Studies in Higher Education*, 25(3):255–265, 2000.

[2] R. Barnett. Learning for an unknown future. *Higher Education Research & Development*, 31(1):65–77, 2012.

[3] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks. In Proc. of the International AAAI Conference on Weblogs and Social Media, 2009.

[4] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[5] M. Borrego and J. Bernhard. The emergence of engineering education research as an internationally connected field of inquiry. *Journal of Engineering Education*, 100(1):14–47, 2011.

[6] E. El-Khawas. Higher education re-formed: Peter scott (ed.): Falmer press, London, 2000. *Higher Education Policy*, 14(1):93–95, 2001.

[7] Z. Fadeeva, Y. Mochizuki, K. Brundiers, A. Wiek, and C. L. Redman. Real-world learning opportunities in sustainability: from classroom into the real world. *International Journal of Sustainability in Higher Education*, 11(4):308–324, 2010.

[8] A. J. Hesketh. Recruiting an elite? employers' perceptions of graduate education and training. *Journal of Education and Work*, 13(3):245–271, 2000.

[9] Y. Jiang, W. Y. S. Lee, and L. Golab. Analyzing student and employer satisfaction with cooperative education through multiple data sources. *Asia-Pacific Journal of Cooperative Education*, 16(4):225-240, 2015.

[10] D. Kember, A. Ho, and C. Hong. The importance of establishing relevance in motivating student learning. *Active Learning in Higher Ed.*, 9(3):249–263, 2008.

[11] R. Lambiotte, J.-C. Delvenne, and M. Barahona. Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint 0812.1770*, 2008.

[12] M. E. Newman. Analysis of weighted networks. *Physical Review E*, 70(5):056131, 2004.

[13] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, Feb 2004.

[14] A. Wiek, L. Withycombe, and C. L. Redman. Key competencies in sustainability: a reference framework for academic program development. *Sustainability Science*, 6(2):203–218, 2011.

[15] A. Wilson. Strategy and management for university development. In *Higher Education Re-Formed*, Falmer Press, pp. 29-44, 2000.

[16] A. Wilson. *Knowledge power: interdisciplinary education for a complex world*. Routledge, 2010.

[17] World Association for Cooperative & Work-integrated Education (WACE). Accessed on 25 Feb 2016, at `www.waceinc.org/global_institutions.html`.

Expediting Support for Social Learning with Behavior Modeling

Yohan Jo[†], Gaurav Tomar[†], Oliver Ferschke[†], Carolyn P. Rosé[†], Dragan Gašević[‡]
[†]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
{yohanj, gtomar, ferschke, cprose}@cs.cmu.edu
[‡]Schools of Education and Informatics, The University of Edinburgh, Edinburgh, UK
dgasev

ABSTRACT

An important research problem for Educational Data Mining is to expedite the cycle of data leading to the analysis of student learning processes and the improvement of support for those processes. For this goal in the context of social interaction in learning, we propose a three-part pipeline that includes data infrastructure, learning process analysis with behavior modeling, and intervention for support. We also describe an application of the pipeline to data from a social learning platform to investigate appropriate goal-setting behavior as a qualification of role models. Students following appropriate goal setters persisted longer in the course, showed increased engagement in hands-on course activities, and were more likely to review previously covered materials as they continued through the course. To foster this beneficial social interaction among students, we propose a social recommender system and show potential for assisting students in interacting with qualified goal setters as role models. We discuss how this generalizable pipeline can be adapted for other support needs in online learning settings.

1. INTRODUCTION

More and more recent work in educational data mining and learning analytics refers to a "virtuous cycle" of data leading to insight on what students need and then improvements in support for learning [17]. An important goal is tightening this cycle to improve learning experience. We are interested especially in social learning, drawing from a Vygotskian theoretical frame where learning practices begin within a social space and become internalized through social interaction. This may involve limited interaction, such as observation, or more intensive interaction through feedback, help exchange, sharing of resources, and discussion.

There are two main contributions of this paper. The first is to propose a pipeline that can expedite the cycle of data infrastructure, learning process analysis, and intervention (Figure 1). Data infrastructure provides a uniform interface for heterogeneous data from social interaction in various platforms, such as connectivist Massive Open Online Courses (cMOOCs) [15], hobby communities, and Reddit communities, where people engage in follower-followee relations, post updates to their account, engage in threaded discussions, and also optionally link in blogs, YouTube videos, and other websites. Learning process analysis aims to analyze students' processes depending on their social network configurations and to identify beneficial kinds of social connections. We developed a probabilistic graphical model that analyzes sequences of behaviors in terms of topics expressed and social media types that students actively engage in over time. Finally, intervention is introduced to foster beneficial social connections among students. We developed a recommender system that matches qualified students to discussions to increase opportunities for them to interact with other peers. The pipeline is iterative such that data from participation is used to create models that trigger interventions in subsequent runs of the course. Data from those later runs can be used to train new and better models in order to improve the interventions, and so on.

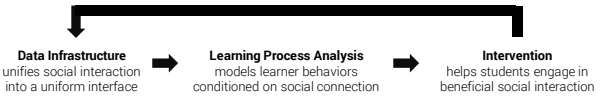


Figure 1: Pipeline for educational data mining in social learning.

Our second contribution is to present findings from an application of the proposed pipeline to data from a social learning environment called ProSolo [12], in order to investigate the positive influence of observing goal-setting behavior. While goal-setting has been intensively researched and proven to be an important self-regulated learning (SRL) practice that often leads to success in learning, the influence of a student's goal-setting behavior on observers has little been investigated empirically. If goal-setting students turn out to be good role models, that is, beneficial to their social peers, we can encourage and help students to make such social connections with goal setters to enhance their learning experience. The usefulness of this effect may be especially desirable in online courses where the number of instructors is limited, or online communities that are not structured like courses, where students are required to take more agency in forging a learning path for themselves within an ecology of resources.

In the remainder of this paper, we first motivate the specifics of our pipeline as situated within the literature. Next, we present our pipeline and its application, along with findings.

2. RELATED WORK

Vygotsky's view of social interaction as a key to learning and Bandura's social learning theory [1] emphasize the importance of interaction to learning. In social contexts, by vicarious learning, students observe external models and learn from those observations even when not actively engaged in interaction [19]. Observation of role models facilitates motivation and self-efficacy for a task [14] and may be associated with positive changes in the observer's behavior [9]. Drawing on this theoretical foundation, the positive impact of social interaction has been investigated in collaborative work [8] and in online courses [11]. Yet, to our knowledge, our work is the first to investigate goal-setting behavior specifically as a qualification of a role model in online learning.

Several data infrastructures have been introduced to aid educational data mining for Massive Open Online Courses (MOOCs). For instance, MOOCdb [18] and DataStage[1], designed to store raw data from MOOCs, consolidate clickstream data from different MOOC platforms in a single, standardized database schema. This allows for developing platform-independent analysis tools, thus enabling analyses that span multiple courses hosted by different MOOC providers with reduced development effort. While these infrastructures focus on behavior data represented by clickstream logs, our proposed infrastructure deeply represents other aspects of student interactions, such as discussion behavior and social relationships, which require the natural language exchange between students.

Analysis of students' learning processes has been a critical topic in education. Our method contributes to the literature on time series behavior modeling. Approaches to learning process analysis differ in the definition of the basic building block, often conceived of as states within a graph. Common building blocks for tutoring systems and educational games include knowledge components [22] and actions [13]. In dialogue settings, it is common to code each utterance according to a coding scheme and analyze the sequence of codes [4]. In a MOOC context, states are often defined as course units [3], course materials [3], or discussions [2]. Such predefined states, however, may not be the ideal units of states, especially in online courses where students can selectively engage in learning resources. Therefore, unsupervised modeling approaches are appealing for the purpose of identifying states that are meaningful indications of student interests obtained in a data-driven way. Our model belongs to the class of Markov models, which have been proposed to learn latent states and state transitions [6, 21].

In MOOCs, a student's learning process is affected by other peers especially through interaction in forums, which offer opportunities to develop communication and community. Hence, social recommendation algorithms can introduce appropriate students to certain discussions for productive interaction. Suggested matches should be appropriate when viewed either from the discussion or student side [16], for example by suggesting a student to participate in discussions based on both the potential benefit of the student's expertise as an asset to the discussions while respecting the limitations of a student's resources for participation in more than a limited number of discussions [20]. Our model can recommend discussions to a student by balancing the benefit of the student's qualification to discussions, her relevance to discussions, and required effort.

[1] http://datastage.stanford.edu/

3. THREE-PART ANALYTICS PIPELINE

Our pipeline is designed to expedite the process of exploiting student data leading to data-driven decision-making for enhancing student learning (Figure 1).

In this pipeline for social learning, the first component is a data infrastructure that maps diverse forms of social interaction into a common structure. This uniform interface allows the subsequent components—learning process analysis and intervention—to apply the same tools to different data, even from distinctly different discourse types, with little modification. Our development of this infrastructure, DiscourseDB[2], represents discourse-centered social interaction as an entity-relation model. Discourses (e.g., forums or social media) and individual contributions in a discourse (e.g., posts, comments, and utterances) are represented as generic containers generalizable to diverse social platforms. DiscourseDB also allows for defining arbitrary relations between contributions, e.g., a "reply-to" relation derived from the explicit reply structure of the platform versus one inferred through some automated analysis process. This flexibility helps the subsequent components of the pipeline avoid data-specific processing. DiscourseDB can store both active and passive activities of individuals, such as creating, revising, accessing, and following contributions, as well as forming social connections with other individuals. DiscourseDB is the key component of our pipeline, based on which the next components perform integrated analyses of discourses and social networking on multiple platforms with reusability.

The second component of our pipeline is analysis of students' learning processes depending on their social connections. The goal is to assess students' needs of support by understanding how learning processes are affected by social interaction and what types of social interactions are helpful to students. Just as Bayesian knowledge tracing enables modeling the learning process from a cognitive perspective and then supporting a student's progress through a curriculum, Bayesian approaches can model learning processes at other levels, including supportive social processes. And similarly, these models can then be used to trigger support for the learning processes in productive ways. Hence, the third component of our pipeline draws upon insights obtained from the analysis to introduce interventions that can help students make beneficial social connections with other peers. We will propose two concrete examples of machine learning techniques for these two components in Section 5 and Section 6 respectively.

4. APPLICATION OF PIPELINE

The remainder of the paper presents an example application of our general pipeline to a specific problem. We propose ex-

[2] http://discoursedb.github.io

ample models for learning process analysis and intervention that can build upon DiscourseDB. After this description we discuss our findings. This section introduces the data set for that exploration.

4.1 Problem and Data

We examine goal-setting behavior as a potential qualification of good role models via learning process analysis and foster social connections with goal setters via recommendation support. Since most MOOCs and informal learning communities lack a measure to identify potentially good role models (e.g., a pretest), increased frequency of effective goal-setting behaviors may serve as an indirect indicator of success, as previous studies showed positive relationships between goal-setting behavior and learning outcomes [5, 23].

The data was collected from an edX MOOC entitled *Data, Analytics, and Learning* (DALMOOC) [12], which ran from October to December 2014. This course covered theoretical principles about learning analytics as well as tutorials on social network analysis, text mining, and data visualization. This MOOC was termed a *dual layer* MOOC because students had the option of choosing a more standard path through the course within the edX platform or to follow a more self-regulated and social path in an external environment called ProSolo. The ProSolo layer allowed students to set their own learning goals and follow other students so that they could view activities and documents that offered clues about how to approach the course productively. While a huge literature on analysis of MOOC data focuses on Coursera, edX, and Udacity MOOCs, other platforms with more social affordances are growing in popularity. In order to serve the goal of identifying support needs and automating support that may be triggered in a social context, it is advantageous to work with data from socially-oriented platforms. We used the log data from ProSolo as our object of analysis, which include students' discussions on ProSolo and their own blogs and Twitter that they identified on their ProSolo profile pages, evidence of students' social connection with each other, and "goal notes," which students can use to set their learning goals in their own words.

We preprocessed discussion data before running our model. First, we filtered course-relevant tweets using the hashtags #prosolo, #dalmooc, and #learninganalytics. We confirmed that the tweets identified as irrelevant by this process have little to do with course activity. Because we are not interested in irrelevant content, we replaced such content with a tag to indicate irrelevant content. In order to prevent topics from being defined in terms of document types, we removed Twitter mentions and "RT" from tweets as well as other function words including URLs from all documents. Descriptive statistics for the data set are listed in Table 1.

4.2 Goal Quality and Social Connection

To categorize the quality of goal-setting behavior of each student, we first annotated each goal note written by students indicating whether it indeed contains a goal or not. 58% of goal notes contained goals. An example goal note is as follows: *"to understand learning analytics and see how these may be useful for my teaching and in particular, my learning resource design/development."* On the basis of this annotation, we categorized students into three classes: (1) goal setters, (2) goal participants, and (3) goal bystanders. Goal setters have goal notes that mention their distal or/and proximal goals. Goal participants have goal notes, all of which are about something other than goals, e.g., experiences or questions. Goal bystanders have no goal notes. Note that the category of a student can change over time. All students start as goal bystanders and may become a goal participant or a goal setter as time passes. A student's *social connection* is then categorized into seven classes: (S1) has already been following a goal setter, (S2) started to follow a goal setter at the current time point (S3) has been following a goal participant (but no goal setter), (S4) started to follow a goal participant at the current time point, (S5) has been following a goal bystander (at best), (S6) started to follow a goal bystander at the current time point, and (S7) follows no one. S2, S4, and S6 mean that a student's social connection improved at the current time point, whereas S1, S3, and S5 indicate that a student remained in the same social connection category as in the previous time point.

Goal notes	62	Tweets (relevant)	715
ProSolo posts	318	Tweets (irrelevant)	25,461
Blog posts	359		
Users	1,729	Social connections	814

Table 1: Descriptive statistics for ProSolo data.

5. LEARNING PROCESS ANALYSIS

Learning process analysis aims to assess students' needs of support. Hence, we model students' behavior and analyze their learning processes as they experience changes in their social connections throughout the course.

5.1 Model

Our model automatically extracts a representation of students' learning processes based on their discussions in a course and their social connections, which may reveal the influence of different configurations within the social space (see our technical report [7] for details). We define the building blocks of learning processes, i.e., states, in terms of discussed topics and the document types used for discussions (e.g. Twitter, blog). Given the sequences of timestamped documents and social connection types for students, our latent Markov model infers a set of states, along with the main topics and document types for each state. The learned topics reflect students' interests, and the document types show how students use different media for different interests. The model also learns transition probabilities between states, conditioned on the social connection category in the source state. This discloses how learning processes differ depending on students' social connection types.

5.2 Findings

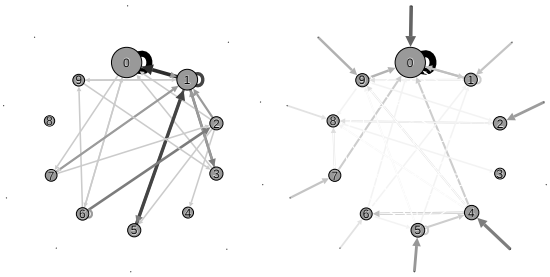
We applied the model to the ProSolo data and examined the correlation between the categories of social connection and learning behaviors. We ran our model with the number of states set to 10 and the number of topics set to 20. We defined the unit of a time point as one week, and if a student had no activity in a certain week, that week was omitted from her sequence.

State	Topics	RelGoalNote	IrGoalNote	Post	Blog	RelTweet	IrTweet
0	Course-irrelevant tweets	0.00	0.00	0.00	0.00	0.00	1.00
1	Concept map, network analysis (Week 9)	0.00	0.00	0.02	0.01	0.18	0.78
2	Social capital (Week 3)	0.04	0.01	0.19	0.30	0.18	0.27
3	Tableau (Week 2), Gephi (Week 3), Lightside (Week 7)	0.01	0.03	0.10	0.28	0.24	0.34
4	Prediction models (Week 5)	0.01	0.02	0.29	0.22	0.10	0.36
5	Data wrangling (Week 2)	0.01	0.01	0.12	0.08	0.26	0.52
6	Visualization (Week 3)	0.05	0.02	0.24	0.47	0.08	0.15
7	Epistemology, assessment, pedagogy (Week 4)	0.05	0.00	0.18	0.22	0.30	0.25
8	Prediction, decision trees (Week 5)	0.02	0.02	0.19	0.40	0.09	0.28
9	Share, creativity (mixed topics)	0.00	0.02	0.12	0.13	0.21	0.52

Table 2: Learned states with their topics and document type distribution (each row sums to 1). (RelGoalNote: goal notes containing a goal, IrGoalNote: goal notes without a goal, Post: posts on ProSolo, Blog: personal blog posts, RelTweet: course-relevant tweets, IrTweet: course-irrelevant tweets)

	Social Connection			
	GS S1+S2	GP S3+S4	GB S5+S6	NO S7
# Time Points	139	315	265	821
% Time Points				
State 0	**0.59****	**0.75**	**0.75**	**0.71**
State 1	**0.17***	**0.10**	**0.03**	**0.04**
State 2	0.05	0.02	0.02	0.04
State 3	**0.04***	**0.00**	0.01	**0.01**
State 4	0.01	0.02	0.03	0.06
State 5	0.05	0.03	0.06	0.05
State 6	0.05	0.02	0.02	0.02
State 7	0.03	0.01	0.03	0.02
State 8	0.00	0.03	0.02	0.02
State 9	0.01	0.04	0.03	0.04

Table 3: Proportion of time points students stay in each state depending on the social connection (each column sums to 1). "$\star\star$" and "$\star$" indicate that GS is significantly different from other categories in bold with $p < 0.01$ and $p < 0.05$, respectively, by Pearson's χ^2 test. GS, GP, and GB each represent either "has been following" or "started to follow" a goal setter, a goal participant, and a goal bystander, respectively. NO means to follow no one.



(a) S1. Following a goal setter (b) S7. Follows no one

Figure 2: State transition patterns. Nodes are states whose size reflects the number of weeks students visit the states. Edges are transitions whose thickness and darkness reflect transition frequency. Edges without a source node represent the probability of being the first state in a learning path.

5.2.1 Learned States

The model learns states with their topics and document type distributions (Table 2). Most states are aligned well with course units covering important course topics. However, State 0 is where students do not participate in course discussion but post course-irrelevant tweets. State 3 is about hands-on practice of software tools across the course, and State 9 covers many side topics. Tweets tend to take a large proportion and goal notes a small proportion in every state due to their relative volumes. Blog posts are actively used for summarizing readings and tutorials, and tweets are used as a means of communicating with lecturers (e.g., State 5). ProSolo posts are most accessible to ProSolo users, so students use them to reveal their opinions and questions.

5.2.2 Students Following Goal Setters

According to the investigation of students' learning processes, based on the number of weeks they spent in each state (Table 3) and state transition patterns (Figure 2), students who follow goal setters show the following positive learning behavior:

Twitter usage: The students following goal setters spend noticeably fewer weeks on irrelevant tweets (State 0).

Participation duration: The topics of the states in which students stay reveal how long they persist in the course. The students following goal setters are more likely to discuss the material taught in the last week (State 1), that is, they are active in the last phase of the course.

Activities of interest: The number of weeks students spend in each state reflects the activities students are interested in. The students following goal setters were more active in hands-on practice (State 3) than other students. Hands-on practice requires higher motivation than merely watching lectures, so these students might have been helped by observation of role models as discussed in the literature [14]. This trend would have not been as clear using predefined states based on course units [3].

Study habits or challenges: Transition patterns may reveal students' study habits or challenges. Figure 2a shows frequent transitions between three states (States 1, 3, and 5) that are associated with materials taught in different weeks. Such transitions may reflect the SRL strategy of activating and applying prior knowledge to the current situation [10].

These positive effects associated with following goal setters are not apparent with other social connection types, e.g., following no one (Figure 2b). This indicates that "who to follow" is more important than simply following someone.

6. INTERVENTION FOR SUPPORT

On the basis of the insights obtained from the previous component, the third component of our pipeline is to offer appropriate support, especially towards fostering beneficial social connections between students. We argue that a recommender system can serve this purpose, by presenting its potential positive impact as assessed on the corpus.

6.1 Model

Our recommender system aims to match qualified students (e.g., goal setters) to discussions so that they can interact with and benefit the discussants through discussions (see our technical report [7] for details). Our model has two steps: relevance prediction and constraint filtering. The relevance prediction step learns the relevance between students and discussions using student- and discussion-related features. The learned relevance reflects students' preferences and tendencies, but may not reflect the ideal matches for fostering learning. The constraint filtering step thus combines the relevance scores with some constraints that foster interaction between qualified students and other students, and finalizes recommendations.

6.2 Findings

Since we have identified positive learning behaviors of students who follow goal setters, we may want to support students by fostering interaction with goal setters. Instead of recommending direct following relations, which are not supported by many learning platforms, we recommend discussions to qualified students so that they can interact with the discussants. We first assess the extent to which students are sensitive to qualified students prior to explicit intervention, and then present the potential added value of our recommendation model.

6.2.1 Students' Awareness of Role Models

Our first step is to assess whether students can identify effective role models in discussion activities (ProSolo posts), by measuring the impact of the information about students' qualifications on the prediction of discussion participation. This task is to infer links between students and discussions that we hid from an observed static snapshot of a network of discussion participation based on observable data. A measured positive impact here would indicate some sensitivity on the part of students to interact with qualified students naturally. We train a predictive model of students' participation in discussions on two thirds of student-discussion pairs. We then predict the discussion participation of the remaining pairs. Our evaluation metric is mean average precision (MAP).

We compared four configurations by varying the information about students' qualifications that is used as feature for relevance prediction. In particular, CAMF uses only basic features, such as the numbers of discussions each student initiated and participated in and each discussion's length, number of replies, and participants. CAMF_G and CAMF_C add information about goal quality and degree centrality, respectively, and CAMF_GC adds both. The evaluation was conducted as a link prediction task, based on the relevance scores predicted in the relevance prediction step. Students' qualification information did not improve link prediction accuracy (Table 4). This means that students are not proactively sensitive to peers' qualifications while participating in discussions, which supports our view that explicit recommendation could be valuable for encouraging students to interact with qualified peers through discussions.

Configuration	MAP	Configuration	MAP
CAMF	0.465	CAMF_C	0.455
CAMF_G	0.438	CAMF_GC	0.439

Table 4: MAP for link prediction.

Configuration	OB	Configuration	OB
GoalPart	1.888	MCCF_G	3.683
HighCent	1.943	MCCF_C	3.770
GoalPart_HighCent	1.873	MCCF_GC	3.656

Table 5: Overall Community Benefit for recommendation.

6.2.2 Recommendation Quality

The recommendation of discussions should be consistent with both the relevance between students and discussions (the relevance prediction step) and constraints for beneficial social connection (the constraint filtering step). To this end, we evaluated recommendation quality on Overall Community Benefit (OB) [7]: the relevance of our recommendations penalized by the burden on the students induced by the recommendations. The higher OB the better.

We tested three configurations by varying the constraints incorporated into the constraint filtering step. MCCF_G requires that every discussion have at least one goal participant or goal setter. MCCF_C requires that every discussion have at least one student whose degree centrality is higher than 0.1. MCCF_GC requires both. In addition, the following configurations were tested as baseline without incorporation into the model. GoalPart filters goal participants or goal setters after making recommendations based on predicted relevance. Similarly, HighCent filters students with degree centrality higher than 0.1. GoalPart_HighCent filters goal participants or goal setters with degree centrality higher than 0.1. Incorporating the constraints about students' goal quality and degree centrality into the model (MCCF_G, MCCF_C, and MCCF_GC) achieved higher OB than the simple filtering approaches (Table 5). That is, our algorithm effectively matches qualified models to relevant discussions in such a way that students in every discussion can interact with qualified models while balancing the load of the models.

7. DISCUSSION

According to our learning process analysis, students benefit from social connections with effective goal setters through ProSolo's follower-followee functionality. They stay longer in the course, engage in hands-on practices, and link materials across the course. This supports the view that goal-setting behavior is a useful qualification for potential role models. According to the discussion participation prediction task, explicit intervention is important for helping students be aware of qualified students and interact with them via discussions. Therefore, we incorporated the information about students' qualifications into our recommendation model as

constraints, successfully matching qualified learning partners to relevant discussions.

This work started from the need for expediting data analysis and analysis-informed support in social learning where students interact with one another via various social media in order to pursue their own learning goals. This expedition builds on DiscourseDB, data infrastructure for complex interaction data from heterogeneous platforms. We proposed a probabilistic graphical model to analyze students' learning processes depending on the state of their social connections, and proposed a recommender system that can improve student support on the basis of the insights obtained from the analysis. This pipeline arguably should allow us to apply the techniques to different learning communities with little effort.

Goal-setting behavior is an important practice in SRL and is known to be difficult for students, so an analysis towards improvement of this skill is arguably valuable. Nevertheless, in this study we have not examined how this behavior influences the domain learning of students. This is due both to the limited data size for our first trial to use ProSolo in MOOCs as well as a lack of learning gain measures. However, the modeling techniques proposed in this paper can readily be applied to other data sets if the requisite data become available. We are also interested in investigating different SRL strategies besides goal-setting in social learning, and how social interaction influences the SRL behaviors of the students. Ultimately, the real value of the work will be demonstrated not with a corpus analysis, as for our proposed recommendation approach, but with an intervention study in a real MOOC. We are working towards incorporating this approach in a planned rerun of DALMOOC.

8. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under grants ACI-1443068 and IIS-1320064, and by the Naval Research Laboratory and Google.

9. REFERENCES

[1] A. Bandura. *Social Learning Theory*. Morristown, N. J.: General Learning Press, 1971.

[2] A. Bogarín and R. Cerezo. Discovering students' navigation paths in Moodle. In *EDM '15*, pages 556–557, 2015.

[3] C. Coffrin, L. Corrin, P. de Barba, and G. Kennedy. Visualizing patterns of student engagement and performance in MOOCs. In *LAK '14*, pages 83–92, Mar. 2014.

[4] A. Ezen-Can, K. E. Boyer, S. Kellogg, and S. Booth. Unsupervised modeling for understanding MOOC discussion forums. *LAK '15*, pages 146–150, 2015.

[5] J. Husman and D. F. Shell. Beliefs and perceptions about the future: A measurement of future time perspective. *Learning and Individual Differences*, 18(2):166–175, Apr. 2008.

[6] Y. Jo and C. P. Rosé. Time Series Analysis of Nursing Notes for Mortality Prediction via a State Transition Topic Model. In *CIKM '15*, 2015.

[7] Y. Jo, G. Tomar, O. Ferschke, C. P. Rosé, and D. Gašević. Expediting support for social learning with behavior modeling. *arXiv:1605.02836*, 2016.

[8] I. Molenaar and M. M. Chiu. Effects of sequences of socially regulated learning on group performance. In *LAK '15*, pages 236–240, Mar. 2015.

[9] E. L. Paluck, H. Shepherd, and P. M. Aronow. Changing climates of conflict: A social network experiment in 56 schools. *Proceedings of the National Academy of Sciences*, 113(3):566–571, Jan. 2016.

[10] P. R. Pintrich. A Conceptual Framework for Assessing Motivation and Self-Regulated Learning in College Students. *Educational Psychology Review*, 16(4):385–407, Dec. 2004.

[11] C. P. Rosé, R. Carlson, D. Yang, M. Wen, L. Resnick, P. Goldman, and J. Sherer. Social factors that contribute to attrition in MOOCs. In *L@S '14*, pages 197–198, Mar. 2014.

[12] C. P. Rosé, O. Ferschke, G. Tomar, D. Yang, I. Howley, V. Aleven, G. Siemens, M. Crosslin, D. Gasevic, and R. Baker. Challenges and Opportunities of Dual-Layer MOOCs: Reflections from an edX Deployment Study. In *CSCL '15*, pages 848–851, 2015.

[13] E. Rowe, R. S. Baker, and J. Asbell-Clarke. Strategic game moves mediate implicit science learning. In *EDM '15*, pages 432–436, 2015.

[14] D. H. Schunk and A. R. Hanson. Peer models: Influence on children's self-efficacy and achievement. *Journal of educational psychology*, 77(3):313–322, 1985.

[15] G. Siemens. Connectivism: A learning theory for the digital age. *International Journal of Instructional Technology and Distance Learning*, 2014.

[16] L. Terveen and D. W. McDonald. Social matching: A framework and research agenda. *ACM transactions on computer-human interaction*, 12(3):401–434, 2005.

[17] C. Thille. Education Technology as a Transformational Innovation. *White House Summit on Community Colleges: Conference Papers*, pages 73–78, 2010.

[18] K. Veeramachaneni, S. Halawa, F. Dernoncourt, U. O'Reilly, C. Taylor, and C. Do. Moocdb: Developing standards and systems to support MOOC data science. *CoRR*, abs/1406.2015, 2014.

[19] P. H. Winne and a. F. Hadwin. Self-regulated learning and socio-cognitive theory. *International Encyclopedia of Education*, pages 503–508, 2010.

[20] D. Yang, D. Adamson, and C. P. Rosé. Question recommendation with constraints for massive open online courses. In *RecSys '14*, pages 49–56, 2014.

[21] J. Yang, J. McAuley, J. Leskovec, P. LePendu, N. Shah, and B. Informatics. Finding Progression Stages in Time-evolving Event Sequences. In *WWW '14*, pages 783–793, Apr. 2014.

[22] C. Zhao and L. Wan. A shortest learning path selection algorithm in e-learning. *Int'l Conference on Advanced Learning Technologies*, pages 94–95, 2006.

[23] B. J. Zimmerman. Goal setting: A key proactive source of academic self-regulation. In *Motivation and self-regulated learning: Theory, research, and applications*, pages 267–295. Erlbaum, 2008.

On generalizability of MOOC models

Łukasz Kidziński, Kshitij Sharma, Mina Shirvani Boroujeni, Pierre Dillenbourg
Computer Human Interaction in Learning and Instruction
École polytechnique fédérale de Lausanne
{lukasz.kidzinski,kshitij.sharma,mina.shirvaniboroujeni,pierre.dillenbourg}@epfl.ch

ABSTRACT

The big data imposes the key problem of generalizability of the results. In the present contribution, we discuss statistical tools which can help to select variables adequate for target level of abstraction. We show that a model considered as over-fitted in one context can be accurate in another. We illustrate this notion with an example analysis experiment on the data from 13 university Massive Online Open Courses (MOOCs). We discuss statistical tools which can be helpful in the analysis of generalizability of MOOC models.

Keywords

Massive open online courses, MOOCs, bias-variance trade-off, generalizability

1. INTRODUCTION

The rapid growth of Massive Online Open Courses (MOOCs) has shown significant impact not only on the education but also on educational research. Over 100 world class universities partner with MOOC platforms to provide free education. Many of these universities, use data analytics to provide indicators to the policy makers, and valuable insights to the teachers and producers.

Researchers from emerging educational fields, such as learning analytics and educational data mining, attempt to make sense from the huge datasets from the MOOC providers (for example Coursera, Edx). These large datasets provide an opportunity to detect the slightest differences in the behaviour which are correlated to the students' performance.

However, the big data involves the risk of misinterpreting the results. The misinterpretations could surface mainly because of two reasons. First, the effect sizes are few orders of magnitude smaller then we used to expect in classical educational psychology studies; and the results are still significant due to the large sample. Second, "black-box" approaches like Support Vector Machines or Neural Networks give us great predictive power of models but do not explain the underlying processes.

Both of these reasons can lead to "overfitting" a model for a given context. Still, the same model can be be accurate in another context as illustrated in Figure 2. Choosing too specific descriptors could lead to the models which precisely describe one student but fail to generalize to new concepts. Too vague descriptors tend to generalize better but inform less about the specifics of the underlying processes. In statistical terminology this is often referred to as the "bias-variance trade-off".

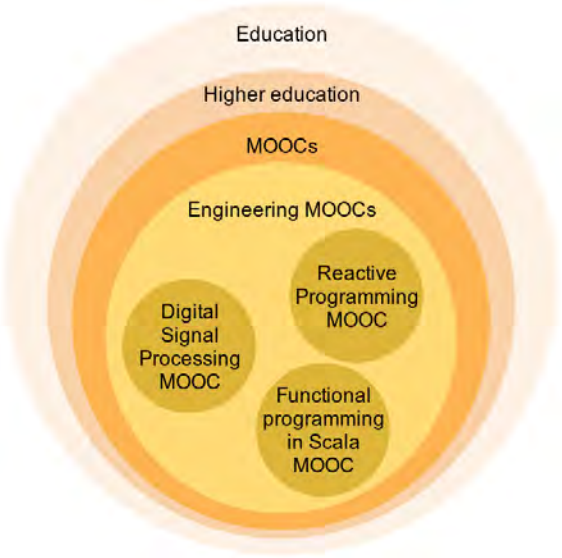


Figure 1: Example of layers to which we can draw conclusions from instances of MOOCs if the generalizability issues are addressed correctly.

The bias-variance trade-off is the central problem in statistical learning. It corresponds to the fact that one cannot minimize both quantities, "bias" and "variance", at the same time. A model with large bias is a smooth model not meant to fit sample points very closely but still captures the general trend in the data. Conversely, a model with large variance (not smooth) varies a lot for similar input parameters in order to fit well to each point in the dataset, often causing the so-called "overfitting".

The objective of this paper is to highlight the potential problem of closed-world context of MOOC research. We discuss techniques for leveraging existing models to more general context. We argue that designing context independent features is crucial for building generalizable models and we illustrate how variable selection process can be enhanced with statistical techniques. We illustrate a statistical technique which can be helpful in the choice of the important variables.

We address the following three research questions:

1. How to measure the extent to which the MOOC research as generalizable?
2. How to leverage predictive models in a MOOC to a broader context?
3. How to improve model's accuracy by restraining the scope of the variables used for prediction purposes?

2. RELATED WORK

2.1 Student Categorization

The common approach for finding generalizable patterns is to classify students into groups. To the best of our knowledge, there exist only a few categorisation schemes, mostly based on what emerges as a pattern of behaviour from MOOC students. These categories are based on the students' motivation [20], engagement patterns [10, 14, 16, 7] or demographics [5, 4].

There are many categorisation schemes depending on the engagement patterns. [10] categorised the students in Completing, Auditing, Disengaging and Sampling students based on their activities which range from watching majority of lectures and submitting all the assignments (Completing) to watching only one or two lectures and no assignment submissions (Sampling). In a connectivist MOOC setting, [14] categorised students into Active (students who adapt well to the connectivest pedagogy), Passive (frustrated ones) and Lurkers (who actively follow the course but do not interact with anyone). Phil Hill first categorised MOOC students into Lurkers (ones who only enrol or sample the course), Active (fully engaged with the course material, quizzes and forums), Passive (only consume the content, did not participate in forums) and Drop-ins (consumed only a part of the course as an Active student) [8]. Later he revised his categories and divided the Lurkers into No-shows and Observers [7].

These schemes are either defined by hard-coded thresholds or by unsupervised learning techniques. For that reason, they remain robust in terms of generalizability within the MOOC's context, but they are hard to generalize outside of it. In this study, we will rather discuss regression than classification/clustering, keeping in mind that similar observations can be done in both contexts.

2.2 Performance and engagement prediction

Student's performance is one of the key metrics analyzed in MOOCs. Many studies chose performance as an indicator for showing the value of the categorization methods. Massive datasets allow us to discover relation between performance and even the smallest factors like the number of pauses during watching a MOOC video or ratio of a video replayed [12]. Performance is also a crucial indicator for policy makers and MOOC practitioners. Reports focus on performance of MOOCs as a function of performance of students [13].

Previous studies on performance often concern a small set of MOOCs [1, 17, 9]. These studies provide insights about a large cohort of students and generalize to another cohorts, however the studies encounter lack of generalizability due to a small sample in the sense of course variability. In other studies, authors used time spent on lecture video, lecture quiz, homework, forum, quiz, assignments to predict students' learning gain [3, 11, 21, 3]. Lauria et al. [11] used the amount of content viewed, forum read, number of posts, assignments and quizzes submitted, to predict the performance and the engagement of the students. Wolff et al. [21] used the temporal clickstream data to predict students' performance.

These studies risk having high bias towards the courses in context and thus might lack the generalizability to be extended to courses with different content and/or courses from different domain. However in the aforementioned works, it is difficult to confirm our claim due to small number of MOOCs being analyzed. An example with generalizable set is shown by [2], where authors used the weekly time series data with 2-, 3-, 4-, and 5- grams to predict the final grades of the students. They experienced issues with the predictive models being generalisable - the model accuracy decreases as the authors used the same course session, to a different session from the same course, to a different course.

3. PROBLEM STATEMENT

In the MOOC context, models with large variance might correspond to the cases where one includes specific information about users, which are characterising only the sample at hand. For example, a model which includes exact timing of actions into account, could fit precisely to the data, since it identifies the user by the time of his actions, but it provides no generizability to new samples. Conversely, models with high bias correspond to situations when one considers general indicators like only the number of forum activities in a MOOC - thus, the model will fit worse to specific users but is more likely to generalize.

In practice, it is impossible to make both variables small, i.e. to retain both good fitness and smoothness. We need to choose the complexity of the model such that the sum of these two quantities is minimised. One could show that for any statistical learning method, the error can be decomposed to variance and bias terms. For a given target value y, predictors x and the estimator $\hat{f}$, the error of the model can be depicted as:

$$\mathrm{E}\Big[\big(y - \hat{f}(x)\big)^2\Big] = \mathrm{Bias}\big[\hat{f}(x)\big]^2 + \mathrm{Var}\big[\hat{f}(x)\big] + \sigma^2, \qquad (1)$$

where σ is the standard deviation of the residuals,

$$\mathrm{Bias}\big[\hat{f}(x)\big] = \mathrm{E}\big[\hat{f}(x)\big] - f(x)$$

and

$$\mathrm{Var}\big[\hat{f}(x)\big] = \mathrm{E}\Big[\big(\hat{f}(x) - \mathrm{E}[\hat{f}(x)]\big)^2\Big].$$

In other terms, bias is the squared distance between the real output $f(x)$ and the average prediction for given x, i.e. the $\mathrm{E}\big[\hat{f}(x)\big]$. The bias gets large whenever the average of predictions x differs highly from $\mathrm{E}\big[\hat{f}(x)\big]$. Conversely, the variance, expressing how do prediction vary from average around x, gets large whenever the variability is high.

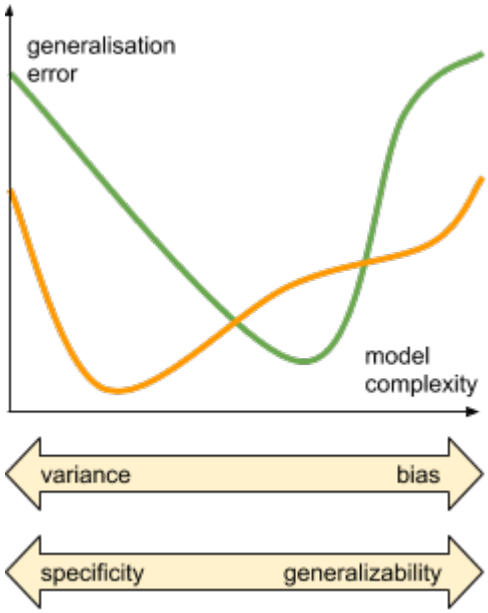


Figure 2: Influence of bias-variance trade-off on the generalzation error - illustrative conceptual drawing.

The ideal model would have both quantities $\mathrm{Bias}\big[\hat{f}(x)\big]^2$ and $\mathrm{Var}\big[\hat{f}(x)\big]$ equal to zero, but, as we mentioned before, it is not practically possible. However, we can control this error, as both quantities depend on the complexity of the model. For example, a linear model with large number of parameters has high variance and thus the error term increases. On the contrary, if one chooses low complexity (small number of variables), the model might have high error due to the high bias. The "best" model is somewhere in the middle, as illustrated by the green curve in Figure 2.

What is often missed in the analysis of the bias-variance plot, is that the error depends also on the context in which we generalize. Particularly in the MOOC context, in Figure 2 the green curve corresponds to generalization to another instance of the same MOOC, whereas the error follows a different pattern (orange curve) if we change the context to another MOOC.

4. MATERIALS AND METHODS

As we focus on the concept of generalizability of models and robustness of variables, we investigate our approach on several different MOOCs. We used data from 13 MOOCs, from EPFL, from both coursera and edX platforms. The dataset contains 1 MOOC which had 3 sessions in 3 consecutive semesters and 2 MOOCs which had 2 sessions in 2 consecutive semesters, as indicated in Table 4.

This setup allows us to investigate several aspects of generalizability. We investigated the fit of a model in correspondance to: 1) the course itself; 2) another instance of the same course; 3) another engineering course.

4.1 Setup

In order to attain a generalizable model, the setup must be consistent between the training data and the test data. Thus, we use the variables which could be defined for all the courses. Additionally, all the scores are normalized to the same range (0 - 100). Since courses have different lengths, we focus only on student activities in the first week. Finally, since 95% of the students did not submit any assignments and significantly bias linear models, we analysed only those students who got at least 1 point as their final grades. Note, that the context we are defining serves mainly as an illustration, thus we choose a relatively simple setup for transparency.

As the measure of performance of a model we take the Normalized Mean Squared Error (`NMSE`), defined as

$$NMSE = \mathrm{Var}(y - \hat{f}(x))/\mathrm{Var}(y),$$

where y is the dependent variable to predict, $\hat{f}$ is the estimator of the relation between y and independent variable x and Var corresponds to the sample variance.

4.2 Example method

In the linear regression, the main source of complexity is due to the number of variables in the model. Classical statistics provide us with robust tools for variable selection, such as ANOVA, Akaike Information Criteria. These techniques are useful for their inferential value, however, they do not guarantee the best generalizability in terms of prediction.

One of the techniques, where the complexity is controlled using a parameter that also affects the performance of the model, is regularized linear regression. In classical statistics, called ridge regression, the standard linear model is extended with an additional, regularizing term. This regularising term controls the parameters of the model with respect to the performance measure based on the prediction, by decreasing the importance of variables which do not account for the prediction.

In particular, given the independent variables $X_1, X_2, ..., X_d$ and the dependent variable Y we build a model minimizing

$$\mathrm{E}\big\|Y - \beta_1 X_1 - \beta_2 X_2 - ... - \beta_n X_d\big\|^2 + \lambda \sum_{i=1}^{k} \beta_i^p, \quad (2)$$

where d is the number of variables, $\beta_1, \beta_2, ..., \beta_d$ are the parameters of the model and $p = 2$.

If λ is large, we put more weight to the sum of βs. Therefore, the number of parameters will be reduced and the model will have a low bias. On the other hand, if λ is small, the model corresponds to linear regression and the variance is high since we use all the variables.

We chose this model for our analysis since it allowed us to control both the bias and the variance with a single parameter λ. Moreover, changing the value of p from 2 to 1 is (2), gives better results in many setups. Hence, we choose to use $p = 1$. The model is known in the machine learning literature as LASSO [19]. The complete algorithm, for those interested, can be found in [19]. Here we are refraining ourselves to the basic description as this is not the main focus of the paper.

4.3 Variables

For illustrating the problem, we chose the students' final grade in the course as the dependent variable. Following are the features that we extracted from the data for modeling this value.

1. **Counts**: We counted different online activities exhibited by the students. 1) *Lectures*: lecture view, lecture re-view, lecture download and lecture re-download; 2) *Quiz*: quiz submission, quiz re-submission, here we differentiated between the quizzes as an exercise, in-video quizzes and the surveys; 3) *Assignments*: assignment submission and assignment re-submission; 4) *Forums*: thread launches, upvotes, downvotes, subscriptions, views, comments and posts.

2. **Delays**: We computed the time difference between the different events in the MOOC structure and students' activities. 1) *First View Delay*: the time difference between the first view or first download of the lecture and the time when the lecture was online; 2) *Overall View Delay*: the average first view delay for all the lecture views and downloads; 3) *Between Lecture Delay*: the time difference between the views or downloads of two different lectures; 4) *Within Lecture Delay*: the average time difference between two views and/or downloads of the same lecture; 5) *First Quiz Attempt Delay*: the time difference between the first submission for a quiz and the time when the quiz was online; 6) *Within Quiz Time*: the time difference between two attempts for the same quiz; 7) *Overall Quiz Attempt Delay*: the average first quiz attempt delays for all the quizzes.

3. **Progress**: We computed the score difference between the two consecutive attempts to the same quiz or the same assignment.

4. **2-way Transitions**: We labeled the different activities as L, A, Q and F for lectures, assignments, quizzes, and forums respectively. Further, we constructed a time-series of the actions and counted how many times the action pairs (for example, AA, AL, AF, LQ, FL, 16 pairs) occur in the time series for each student.

5. **3-way Transitions**: using the same time series, as to compute the 2-way transitions, we counted how many times the action triples (for example, AAA, FAL, QAF, LLQ, FLL, 64 triples) occur in the time series for each student.

5. RESULTS

Using the variables, defined in the Section 4.3, we illustrate the setup for modelling the data. As we mentioned in the Section 4.1, we considered only the activities from the first week of the courses and from those students who scored at least 1. We would also like to emphasize here that the main aim of this contribution is not to present a model that has the least error, but to show how we can build generalizable models taking into account the bias-variance trade off.

In the proposed setup, we demonstrate how generalizable a model is to: 1) the students from the same course (separate test set of 20% of observations), 2) the students from another instance of the same course, 3) to a different course.

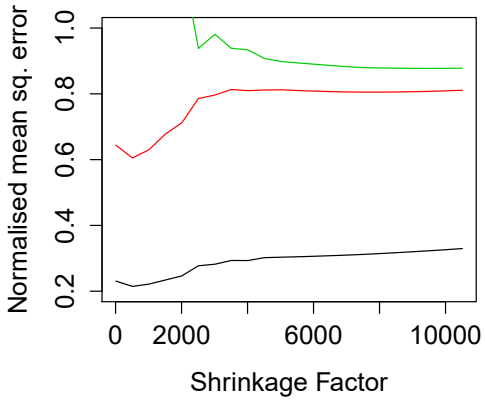


Figure 3: Prediction error (NMSE) for the test samples, for the different values of the shrinkage factor λ in (2), using all the variables.

First, we analyze the model fit to the first session of the *Numerical Analysis* course and test it on: itself, another session of *Numerical Analysis* and *House Water Treatment Systems* a course from a different domain. We illustrate the results in Figure 3. We observed that the model which had highest predictive power on the test set in the session 1 (black curve) has the worse predictive power for another instance of the same course (red curve), but still performs well. The optimal shrinkage factor (λ in equation (2)) turns out to be close to 0 in both cases. This shows that almost all the variables we introduced are included in the model. We could conclude that the model generalizes to another instances of the same course.

However, as we hypothesized, the full model did not fit at all to a course from a different domain. Only with a large value of the shrinkage factor, which removed 97 variables

from the model, we obtain a model with some informative value for a course from another domain. Furthermore, the errors become similar for all the courses, illustrating that the model has lower variance. It generalizes better to another course but it lost its fit to the Numerical Analysis course.

We conducted the identical analysis (see Table 1) on all the courses mentioned in the dataset. In all the cases, generalizability to another course required significant decrease in the complexity, using the shrinkage factor. Removing certain variables from the model turns out to be crucial for the performance. Since we started with 134 variables, to further analyze the ability to generalize, we restricted ourselves to a simpler case with the first three (counts, delays and progress) groups of variables introduced in Section 4.3.

The same patterns were observed in this simpler case. The optimal model for prediction in the same instance and in another instance of the course have the lowest error if the complexity (variance) is high. However, the model with such a high complexity exhibits poor performance in another course, from the same domain, i.e. the linear optimization.

As hypothesized, variables which were removed by LASSO, are course-structure dependent. The most generalizable models contain the variables related to the lecture, forum and quiz activities. These variables provide the required generalizability to the model and hence we observe that as we increase the shrinkage factor, the predictive power of the model increasingly became similar for the different courses.

6. CONCLUSION

We demonstrated through examples that in the terms of bias-variance trade-off, achieving both the specificity and generalizability is not possible while modelling student behaviour. Through the statistical methods available, one can only achieve one of the two goals, or find an optimum solution that is specific to one course and only reveals the surface learning behaviour of the students from a course from another domain, or vice verse.

Similar validation framework, analysing fitness in the same course, another session of the same course and another course was previously introduced [2] in literature. Results from this work are equivalent to ours with some predefined and fixed complexity parameter. In our work we show that practitioners can modulate the complexity and generalizabiity by selecting a subset of variables.

Previous works, have small sample size in terms of number of MOOCs. It is therefore difficult to assess their generalizability. For example, Social Network Analysis (as shown by [18, 15, 6]) is based on the motivation of the student - if the students are sharing the exact answers (or revealing them in some other ways) forum view can play a big role in achievement. Clickstreams (as shown by [21]) in a video are highly dependent on the content. Finally, from the methodological perspective generalizability is also a design choice - for example - if we choose a smaller number of clusters in unsupervised learning, we may obtain more robust results (smaller variance higher bias).

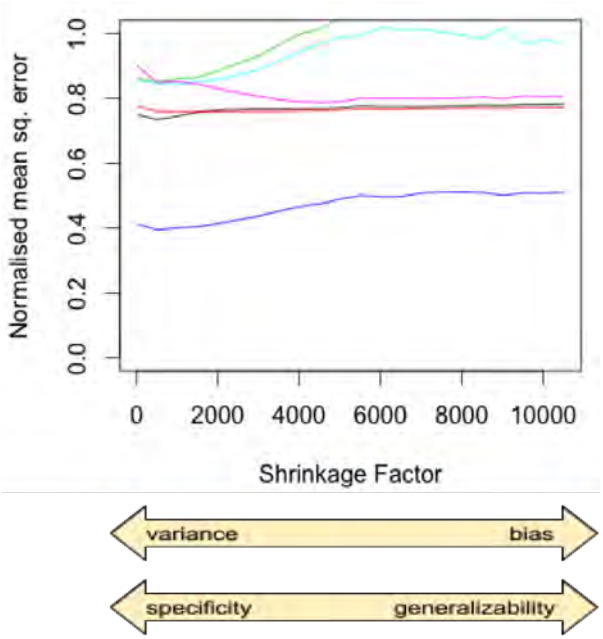


Figure 4: Illustration of bias-variance trade-off from engineering courses. Prediction error (NMSE) for the test samples, for the different values of the shrinkage factor λ in (2)

7. DISCUSSION

Our goal was to illustrate the generalizability issue which we encounter in any machine learning or learning analytics setups. We did not compare multiple algorithms, but we used a simple one to support our claims. It is worth mentioning that the same phenomenon is encountered in any other machine learning method.

Moreover, the same analysis can be performed with any regularized regression algorithms, i.e., consisting a parameter to control the complexity of the model, like SVM, logistic regression, neural networks, etc. In each of these methods regularization selects the optimal sets of parameters.

Finally, the choice of the feature set should be based on the desired outcome of modelling student behaviour in a MOOC. If the goal is to attain high predictability in a small variety of courses, one could choose to include course-structure related variables. On the other hand, if the modelling requirement is to have a decent genralizability over a wide variety of courses, one has to compromise the predictability over a set of courses and select only the course-structure-independent variables.

8. REFERENCES

[1] L. Breslow, D. E. Pritchard, J. DeBoer, G. S. Stump, A. D. Ho, and D. T. Seaton. Studying learning in the worldwide classroom: Research into edxâĂŹs first mooc. *Research & Practice in Assessment*, 8(1):13–25, 2013.

[2] C. Brooks, C. Thompson, and S. Teasley. A time series

Table 1: Results from the identical analysis done on all the other courses as shown in Figures 3. The courses with N/A in the second column had only one session. The errors reported are NMSE. The values in the perenthesis are the optimal shrinkage factors in given context.

Course Name	Testing on the same course	Testing on other session	Testing on different course
Digital signal processing	0.76 (10)	0.99 (10)	0.35 (2010)
Geomatics	0.67 (10)	N/A	0.35 (2010)
House water treatment systems	0.58 (10)	N/A	0.48 (510)
Linear optimisation	0.67 (10)	N/A	0.36 (3010)
Mechanics	0.68 (10)	N/A	0.59 (1010)
Sanitation	0.80 (510)	N/A	0.73 (1010)
Structures	0.95 (10)	0.93 (2010)	0.84 (4510)
Micro-controllers	0.35 (2010)	0.35 (2010)	0.35 (2010)

interaction analysis method for building predictive models of learners using log data. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 126–135. ACM, 2015.

[3] J. Champaign, K. F. Colvin, A. Liu, C. Fredericks, D. Seaton, and D. E. Pritchard. Correlating skill and improvement in 2 moocs with a student's time on tasks. In *Proceedings of the first ACM conference on Learning@Scale conference*, pages 11–20. ACM, 2014.

[4] G. Christensen, A. Steinmetz, B. Alcorn, A. Bennett, D. Woods, and E. J. Emanuel. The mooc phenomenon: who takes massive open online courses and why? *Available at SSRN 2350964*, 2013.

[5] J. DeBoer, G. S. Stump, D. Seaton, and L. Breslow. Diversity in mooc students? backgrounds and behaviors in relationship to performance in 6.002 x. In *Proceedings of the Sixth Learning International Networks Consortium Conference*, 2013.

[6] A. Ezen-Can, K. E. Boyer, S. Kellogg, and S. Booth. Unsupervised modeling for understanding mooc discussion forums: a learning analytics approach. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 146–150. ACM, 2015.

[7] P. Hill. Emerging student patterns in moocs: A graphical view, 2013.

[8] P. Hill. The four student archetypes emerging in moocs. *E-Literate. March*, 10:2013, 2013.

[9] S. Jiang, A. Williams, K. Schenke, M. Warschauer, and D. O'dowd. Predicting mooc performance with week 1 behavior. In *Educational Data Mining 2014*, 2014.

[10] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 170–179. ACM, 2013.

[11] E. J. Lauría, J. D. Baron, M. Devireddy, V. Sundararaju, and S. M. Jayaprakash. Mining academic data to improve college student retention: An open source perspective. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 139–142. ACM, 2012.

[12] N. Li, Ł. Kidziński, P. Jermann, and P. Dillenbourg. Mooc video interaction patterns: What do they tell us? In *Design for Teaching and Learning in a Networked World*, pages 197–210. Springer International Publishing, 2015.

[13] A. McAuley, B. Stewart, G. Siemens, and D. Cormier. The mooc model for digital practice. 2010.

[14] C. Milligan, A. Littlejohn, and A. Margaryan. Patterns of engagement in connectivist moocs. *MERLOT Journal of Online Learning and Teaching*, 9(2), 2013.

[15] W. C. Paredes and K. S. K. Chung. Modelling learning & performance: a social networks perspective. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 34–42. ACM, 2012.

[16] T. Petty and A. Farinde. Investigating student engagement in an online mathematics course through windows into teaching and learning. *Journal of Online Learning and Teaching*, 9(2):261–270, 2013.

[17] S. Rayyan, D. T. Seaton, J. Belcher, D. E. Pritchard, and I. Chuang. Participation and performance in 8.02 x electricity and magnetism: The first physics mooc from mitx. *arXiv preprint arXiv:1310.3173*, 2013.

[18] D. Rosen, V. Miagkikh, and D. Suthers. Social and semantic network analysis of chat logs. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*, pages 134–139. ACM, 2011.

[19] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[20] J. Wilkowski, A. Deutsch, and D. M. Russell. Student skill and goal achievement in the mapping with google mooc. In *Proceedings of the first ACM conference on Learning@ scale conference*, pages 3–10. ACM, 2014.

[21] A. Wolff, Z. Zdrahal, A. Nikolov, and M. Pantucek. Improving retention: predicting at-risk students by analysing clicking behaviour in a virtual learning environment. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 145–149. ACM, 2013.

Closing the Loop with Quantitative Cognitive Task Analysis

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15201
koedinger@cmu.edu

Elizabeth A. McLaughlin
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15201
mimim@cs.cmu.edu

ABSTRACT

Many educational data mining studies have explored methods for discovering cognitive models and have emphasized improving prediction accuracy. Too few studies have "closed the loop" by applying discovered models toward improving instruction and testing whether proposed improvements achieve higher student outcomes. We claim that such application is more effective when interpretable, explanatory models are produced. One class of such models involves a matrix mapping hypothesized (and typically human labeled) latent knowledge components (KCs) to the instructional practice tasks that require them. An under-investigated assumption in these models is that both task difficulty and learning transfer are modeled and predicted by the same latent KCs. We provide evidence for this assumption. More specifically, we investigate the data-driven hypothesis that competence with Algebra story problems may be better enhanced not through story problem practice but through, apparently task irrelevant, practice with symbolic expressions. We present new data and analytics that extend a prior close-the-loop study to 711 middle school math students. The results provide evidence that *quantitative cognitive task analysis* can use data from task difficulty differences to aid discovery of cognitive models that include non-obvious or hidden skills. In turn, student learning and transfer can be improved by closing the loop through instructional design of novel tasks to practice those hidden skills.

Keywords

Cognitive task analysis, cognitive model, transfer, knowledge components, close-the-loop experiment

1. INTRODUCTION

As the field of Educational Data Mining (EDM) strives for technical innovation, there is risk of losing the "E" in "EDM", that is, of not making a clear link to the "Educational" in "Educational Data Mining". Connected with this concern is the temptation to evaluate EDM research only in terms of predictive accuracy and not place value on interpreting the resulting models for plausibility and generalizable insights. While it is possible to use uninterpretable or "black box" predictive models in educational applications (e.g., [1]), interpreting model results is an important step toward improving educational theory and practice for three reasons: 1) for advancing scientific understanding of learning or educational domain content, 2) for generalization of models to new data sets (cf., [19]), and 3) for gaining insights that lead to improved educational technology design.

Whether an educational application of EDM is through a black box model or mediated by data interpretation, the most important, rigorous, and firmly grounded evaluation of an EDM result is whether an educational system based on it produces better student learning. Such an evaluation has been referred to as "closing the loop" (e.g., [16]) as it completes a "4d cycle" of system **d**esign, **d**eployment, **d**ata analysis, and **d**iscovery leading back to design. The loop is closed through an experimental comparison of a system redesign with the original system design.

Use of the "close the loop" phrase, in our writing, goes back at least to [12]. Early examples of data-driven tutor designs, that is, of a close-the-loop experiment, can be found in [13] which tested a tutor redesign based on discoveries from data originally published in [17] and in [4], which was based on data analysis [5]. It is notable that a systematic process for going from data to system redesign was not articulated in this early work, but has been increasingly elaborated in more recent writings [especially 16].

This paper further specifies a particular class of analytic methods, namely *quantitative cognitive task analysis* methods, and how to use them to close the loop. The output of a cognitive task analysis (CTA) is a model of the underlying cognitive processing components (so-called knowledge components or KCs) that need to be learned to perform well in a task domain. Quantitative CTA uses data on task difficulty and task-to-task learning transfer to make inferences about underlying components.

1.1 Cognitive Task Analysis

In general, Cognitive Task Analysis (CTA) uses various empirical methods (e.g., interviews, think alouds, observations) to uncover and make explicit cognitive processes experts use and novices need to acquire to complete complex tasks [3]. Various representations of the resulting cognitive model (e.g., goal trees, task hierarchies, if-then procedure descriptions) are used to design or redesign

instruction. Close-the-loop experiments in different domains demonstrate that students learn more from instruction based on CTA than from previously existing instruction (e.g., medicine [23]; biology [8]; aviation [20]). These results come from CTAs using qualitative research methods that are costly and substantially subjective.

Quantitative CTA methods provide greater reliability and are less costly (though ideally used as a complement to qualitative CTA). An early close-the-loop study [13] based from a Difficulty Factors Assessment (DFA) showed that algebra students are better at solving beginning algebra story problems than matched equations. In a controlled random assignment experiment, the newly designed instructional strategy was shown to enhance student learning beyond the original tutor. Besides DFA, automated techniques can further reduce human effort and can be used on large data sets. An early example used learning curve analysis to identify hidden planning skills in geometry area [16] that resulted in tutor redesign. In a close-the-loop experiment comparing the original tutor to the redesigned tutor, students reached mastery in 25% less time and performed better on complex planning problems on the post-test. Further research [15] has shown how a search algorithm (e.g., Learning Factors Analysis) can generate better alternative cognitive models.

A key assumption behind DFA is that significant differences in task difficulty can be used to make non-obvious (sometimes counter-intuitive) inferences about underlying cognitive components and, in turn, these components help predict learning transfer and guide better instructional design. Similarly, statistical models of learning, including both logistic regression and Bayesian Knowledge Tracing variations, also tend to assume that both task difficulty and learning transfer can be predicted using the same KC matrix.

Recent work explored this connection [18] and found, across 8 datasets, that statistical models that use the *same* KC matrix to predict task difficulty *and* learning transfer produce better results than models that use *separate* matrices (item vs. KC). A key goal of this paper is to further investigate this difficulty-transfer linkage claim by extending evaluation of it through close-the-loop experimentation.

1.2 Illustrating Quantitative CTA

Consider the problems in Table 1 and try to answer the following question before reading on. Assuming the goal of instruction is to improve students' skill at translating story problems into algebraic symbols (e.g., translating the 2_step story in the first column of Table 1 into "62+62-f"), which will yield better transfer of learning: practice on 1_step story problems (columns 2 and 3) or practice on substitution problems (column 4)? Note that in the close-the-loop experiment we ran, similar multiple matched problem sets were created. A different problem set was used for practice than was used for transfer. For example, students who saw the 2-step problem in Table 1 as a transfer post-test item would not see the associated 1_step or substitution problems from Table 1 as practice problems. So, again, which yields better transfer to 2_step problems, practice on 1_step or substitution?

If you answered that practice on the 1_step story problems will better transfer to 2_step story problems, you are in good company as learning commonalities underlying problem formats (i.e., deep features) is a known factor in aiding analogy and transfer [9; 10]. But, the following quantitative analogy cognitive task analysis suggests a different answer.

Table 1. Examples of problem variations and their solutions.

2_step	1_step	1_step	substitution
Ms. Lindquist teaches 62 girls. Ms. Lindquist teaches ***f fewer boys*** than girls. Write an expression for how many students Ms. Lindquist teaches.	Ms. Lindquist teaches 62 girls. Ms. Lindquist teaches ***b boys***. Write an expression for how many students Ms. Lindquist teaches.	Ms. Lindquist teaches 62 girls. Ms. Lindquist teaches ***f fewer boys*** than girls. Write an expression for how many boys Ms. Lindquist teaches.	Substitute 62-f for b in 62+b Write the resulting expression.
62+62-f	62+b	62-f	62+62-f

Using DFA, [11] explored the struggle beginning algebra students have with translating story problems into symbolic algebra expressions. A common belief is that story problems are hard due to comprehending the story content. However, two results indicate that comprehension is not a major roadblock. First, students are better able to solve 2_step problems when given a value (e.g., answering 116 when f is given as 8 in the 2_step story shown in Table 1) than when asked to write the symbolic expression (e.g., 62+62-f or even 62+62-8) [11]. Second, students do not do better when given explicit comprehension hints of the needed arithmetic operations than they do on 2_step symbolization problems without hints [11]. If comprehension is not the key challenge, perhaps production of the target algebraic symbols is. Their results show students perform consistently better (62% vs. 40%) symbolizing both 1_step problems (e.g., producing 62+b and 62-f for the 1_step problems in Table 1) than on 2_step problems (e.g., producing 62+62-f for the 2_step story problem in Table 1).

These results suggest inferences about unobserved or "hidden skills" that are needed to translate 2_step stories into symbolic expressions such as learning how to put one algebraic expression inside another (e.g., as the one- operator expression 40m is inside the two-operator expression 800- 40m). The results are consistent with a need for skills that extend the implicit grammar for generating expressions for 1_step symbolization to recursive structures (e.g., "expression => expression operator quantity" and "expression => quantity operator expression"). Furthermore, they suggested that practicing non-contextual substitution problems (see last column of Table 1) should help students (implicitly) learn the desired recursive grammar structures and the corresponding production skills for constructing more complex expressions.

1.3 Analysis Methods

Our first analysis explores how much substitution practice transfers to story symbolization. We pursue this question with respect to broad outcomes and learning processes. This analysis replicates the high level analysis of the prior study (2008-09) [14] with a full dataset accumulated across four school years (2008-12). Our second analysis probes, more specifically, the question of the cognitive model link between task difficulty and learning transfer that underlies quantitative cognitive task analysis and, more generally, adaptive tutoring models like Bayesian Knowledge Tracing. Practically, the theoretical claim that learning transfer can be inferred from task difficulty data suggests that we can design instruction that produces better transfer of learning using models built from difficulty data (which is easier to collect).

Our third analysis examines whether statistical models of the learning process data support conclusions drawn from the outcome data. Does learning curve analysis indicate whether and how tasks (e.g., substitution problems) designed to isolate practice of CTA-discovered hidden skills (e.g., recursive grammar) transfer to complex tasks that theoretically require these skills (e.g., 2_step story problems)?

2. METHOD

The original 2008-09 study [14] and current close-the-loop study were run with middle school students as part of a math course. In the original study, students were randomly assigned to either a substitution practice condition (N=149) or 1_step story practice condition (N=154). Since then, additional data with random student assignment was collected over three school years from 2009-12 (N=234 for substitution practice, N=174 for 1_step story practice) using the same problem set in ASSISTments. As previously described [14], the study involved a pre-test, instruction, and post-test. For the substitution condition, substitution problems were embedded as instruction interleaved with 2_step story problems (posttest). For the 1_step condition, 1_step problems were used as instruction interleaved with the same 2_step story problems. The pretest for a given version and order was the same for both conditions. Order was determined by difficulty of 2_step problems from a pilot study and included a sequence of 2_step problems from easy to hard or hard to easy.

Small changes were made to the automated scoring to give better feedback on unusual but arguably correct answers (e.g., d60 instead of 60d). For consistency in scoring, manual corrections made to the 0809 dataset [14] were combined with the corrections to the 0912 dataset and automatically applied to every answer in the combined dataset (0812).

3. RESULTS AND DISCUSSION

3.1 High Level Transfer

In our first study [14], we reported significant main effects for condition and order while controlling for pretest, and no significant two-way or three way interaction effects when version was added as an independent variable. In the new study, we add a fifth factor for when the data was collected (i.e., from 0809 or from later years 0912). Most importantly, in a full five-factorial ANCOVA (in R with pretest as the covariate), we found a main effect for condition ($F(1,679) = 4.5$, $p < 0.05$, $d = .21$). Main effects were also found for pre-test ($F(1,679) = 235.3$, $p < 0.001$), order ($F(1,679) = 117.8$, $p < 0.001$), and version ($F(1,679) = 19.8$, $p < 0.001$), but study year was insignificant. Significant two-way interactions were found for pre-test and condition ($F(1,679) = 4.05$, $p < 0.05$), pre-test and order ($F(1,679) = 18.69$, $p < 0.001$), and order and year ($F(1,679) = 10.77$, $p < 0.01$). No other higher- level interactions were significant (all $p > 0.05$).

The pre-test by condition interaction is a consequence of the substitution treatment having a greater effect for students with higher pre-tests. Based on a median split of pre-test scores, students with a higher pre-test, showed greater benefits of substitution practice (52% posttest) over 1_step practice (44%). In contrast, students with a lower pre-test show less benefit of substitution practice (24% posttest) over 1_step practice (20%). This interaction is theoretically consistent with the cognitive task analysis in that students who cannot generate symbolizations for 1_step problems (e.g., 800-y and 40x) will not have the raw material they need to compose 2_step expressions (e.g., 800-40x). Figure 1 illustrates the interaction. Substitution practice produces transfer to story problem symbolization for the 82% of students (580 of 711) with pre-tests of at least 40%. For the 18% of students without 1_step story skills (below 40% on the pre-test), substitution practice does not provide a benefit.

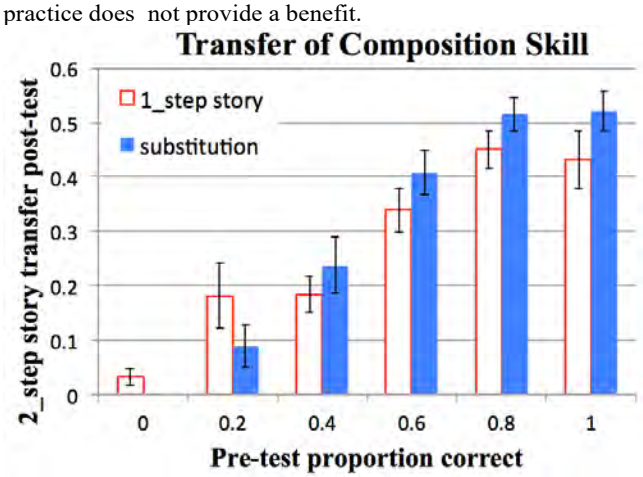


Figure 1. The benefit of substitution practice for symbolizing 2_step story problems is present for the 82% of students with some incoming competence in 1_step story symbolization (at least 40% correct).

The two other reliable interactions in the ANCOVA are not of theoretical significance, but we report them for completeness. The pre-test by order interaction is manifest in that the difference between high and low pre-test students is bigger on the easier post-test problems (63% - 31% = 32%), which appear in the hard-to-easy order, than on the harder post-problems (38% - 10% = 28%), which appear in the easy-to-hard order. The order by year interaction is a consequence of students in the 0912 school years showing more sensitivity to the order manipulation than students in the 0809 school year, such that they do relatively better on the easy problems (46% vs. 41%), but worse on the hard problems (24% vs. 30%).

3.2 Difficulty Reliably Predicts Transfer

In this analysis, we more precisely test the following general logic: If difficulty data indicates a hidden skill that makes an important task hard, then inventing new practice tasks to isolate practice of that hidden skill will transfer to better learning of that hard task. The specific version of the logic in this domain is: If the hard part of symbolizing a two operator story problem is in composing symbolic expressions, then practice on substitution problems should transfer to better performance on story problem symbolization. Our data set affords an interesting opportunity to more precisely test this logic because the difficulty data we have indicates hidden skills for some problem types, but not others. A precise application of the "hidden-skill-transfer" logic stated above is that we should see the predicted transfer for those problem types in which the hidden skill is indicated by the difficulty data. For the other problem types, there should be no reliable transfer.

We used the current data to reevaluate the "composition effect" [11]. This analysis is shown in Table 2 where task difficulty and transfer results are shown for each of the eight problems. Consider the row for the *class* problem (referred

to as "*students"* in the data file), which is illustrated in Table 1. The answer for the 2_step story and substitution problems, namely 62+62-f, is shown in the second column. The third and fourth columns show the proportion correct on the 1_step story problems, (.75 for the "a" step with the answer 62+b and .70 for the "b" step with the answer 62-f). The fifth column (labeled a*b) shows the probability of getting both of these steps correct, computed here as the product of the proportion correct on each step, .53 = 75*.70. This value is the baseline for the composition effect.

The sixth column is the proportion correct on the 2_step story problem, 0.13. This value was computed from student performance on the pre-test for both conditions and the post-test for the 1_step practice condition. We did not use the post-test for the substitution practice condition to estimate the composition effect as the theory predicts that substitution practice should reduce that effect.

A composition effect is indicated when students are less likely to correctly symbolize a two operator story than to correctly symbolize both of the matched one operator stories. The seventh column displays this difference (.40 = .53 - .13 for the *class* problem). The eighth column shows the estimated conditional probability that students can compose a single two-operator expression (e.g., 62+62-f) given they have correctly formulated the two source one-operator expressions (e.g., 62+b and 62-f). Since p(2_step) = p(a*b) * p(2_step | a*b), we get p(2_step | a*b) = p(2_step)/p(a*b), thus for the *class* problem p(2_step | a*b) = .13/.53 = .25. The lower this value, the bigger the composition effect.

The important feature to note about values in the composition effect columns is that they indicate there is no composition effect for the *cds* and *mcdonalds* problems (see the last two rows). Both are relatively well-practiced forms, the 5h-7 for *mcdonalds* is a high frequency linear form (i.e.,

Table 2. Composition effects are found for all but the bottom two problems

Problem name	2_step solution	1_step (a)	1_step (b)	a*b	2_step	Composition Effect		Subst transfer
						a*b - 2_step	2_step/(a*b)	
trip	550/(h-2)	0.65	0.78	0.51	0.11	0.40	0.22	0.08
class	62+62-f	0.75	0.70	0.53	0.13	0.40	0.25	0.12
jackets	d-1/3*d	0.58	0.54	0.29	0.16	0.13	0.56	-0.02
sisters	(72-m)/4	0.71	0.63	0.45	0.32	0.13	0.72	0.15
rowboat	800-40m	0.75	0.55	0.38	0.28	0.10	0.73	0.07
children	(972+b)/5	0.66	0.75	0.5	0.38	0.12	0.76	0.09
cds	5*12*c	0.71	0.74	0.52	0.52	0.00	1.00	0.14
mcdonalds	5*h-7	0.66	0.85	0.56	0.72	-0.16	1.29	-0.06

mx+b) and the *cds* form 5*12*c involves a repetition of the same operator which can be treated as a 1-operator solution, namely, 60c (as 17% of students did). Students may have specialized knowledge for producing these forms that do not require general recursive grammar knowledge.

The final column (Subst transfer) shows how much substitution practice transferred to 2_step symbolization as computed by the difference in post-test scores on each problem for the two experimental groups.

To test the hidden-skill-transfer hypothesis, we expect the *cds* and *mcdonalds* problems to show less transfer and the other problems to show more. While this is not strictly the case (*cds* shows transfer and *jackets* does not), there is a trend here that is illustrated in Figure 2. It shows the relationship between difficulty variation in the composition process and variation in the amount of transfer produced by substitution practice in the close-the-loop experiment. To better highlight the point, the graph shows the data from the 353 students at or above the median on the pre-test -- the ones for which improvement in composition skills should produce better post-test performance on 2_step story problems requiring such skills.

Consistent with the hidden-skill-transfer hypothesis, there is no transfer benefit (first two bars in Figure 2) for the two problem forms with no composition effect (*mcdonalds* and *cds*). There is large transfer effect for the three problems (*trip*, *sisters*, and *children*) involving parentheses (last two bars), which present greater challenges for composing expressions and the need for students to acquire more complex implicit grammar structures for generating correct parenthetic expressions. There is an immediate transfer effect for the three problems (*class, jackets,* and *rowboat)* not involving parentheses (middle bars), consistent with the fewer composition skills required. Note that success on these problems is oddly lower overall. We return to this point in the learning curve analysis where we do some search for new difficulty factors

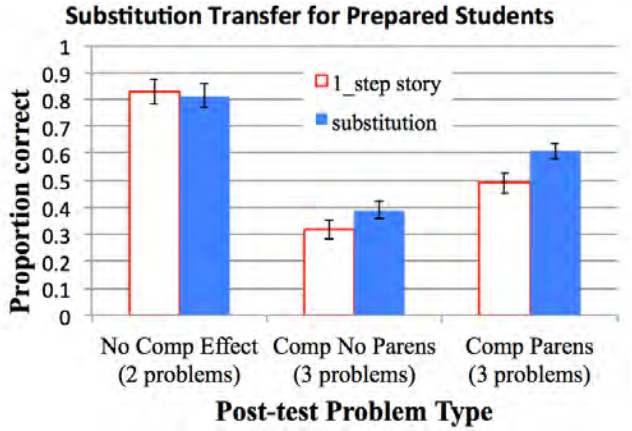


Figure 2. Transfer is limited to the problems that show a composition effect in task difficulty comparisons.

and hypothesize a new hidden skill that could be pursued in future close-the-loop instructional design. These results add to prior evidence [18] supporting the hypothesis that differences in task difficulty and transfer effects are observable manifestations of the same underlying KCs.

3.3 Learning Curve Analysis

As a visual representation of student performance data over time (i.e., as opportunity increases, error rates are expected to decrease), learning curves can be used to explore areas of student difficulty and transfer of learning [21]. Following this prior work, we used the statistical model for learning curve prediction built into DataShop (see PSLCDataShop.org): The Additive Factors Model is a logistic regression model that generalizes Item Response Theory by having latent variables for knowledge component difficulty in place of item difficulty and by adding a third growth term, a knowledge component learning rate, in addition to the student proficiency and knowledge component difficulty terms. We evaluate four different knowledge component models in terms of their prediction fit to all of the test and instructional items each student experienced. For our metrics, we use root mean squared error (RMSE) averaged over 20 runs of 3-fold item-stratified and student-stratified cross validation. Given the focus on understanding the difficulty and transfer characteristics of the task environment, we put particular value on predictive generalization across items (as item stratification achieves by randomly putting all data on each item in the same fold) but also report the predictive generalization across students (as student stratification achieves by randomly putting all data on each student in the same fold).

The results of a learning curve analysis are shown in Table 3. The first row displays a simple baseline no-transfer model that treats each problem type (2_step, 1_step, and substitution) as requiring a different knowledge component (KC). The second row displays a substitution transfer model that introduces transfer between substitution problems and 2_step problems by having a recursive grammar KC common to both problems. The 2_step problems have an additional KC for comprehending the story and the 1_step problems have a different unique KC. As shown in the last columns, this substitution transfer model produces a reduction in RMSE on the item stratified cross validation, down to 0.426 from 0.429. This small change is associated with a small change in the models and changes at this level (in the thousandths) have proven meaningful in producing a prior close-the-loop improvement [16]. This close-the-loop study provides further evidence that small prediction differences can be associated with significant learning gains.

Corresponding with the discussion above regarding the unique challenges of solutions requiring parentheses, the paren-enhanced model (third row in Table 3) adds a parenthesis KC to the 2_step and substitution versions of the *trip, sisters,* and *children* problems. Surprisingly, this model does not improve the item generalization (0.428 > 0.426), though it does improve student generalization (0.473 < 0.477). The predictions of this model fail to account for the variance in difficulty of the non-parentheses problems.

As mentioned above, we were surprised that a couple of the non-parentheses problems posed great difficulty. In particular, the *class* (62+62-f) and *jackets* (d-1/3d) problems were quite hard (13% and 16% correct before substitution instruction). We hypothesized the difficulty of these problems was due to a quantity being referenced twice in the solution expression (i.e., 62 in the *class* problem and d in the *jackets* problem). To test this hypothesis we built the double-ref-enhanced model (fourth row in Table 3) by adding a double-ref KC to the paren-enhanced model on both of the 2_step and substitution versions of the *class* and *jackets* problems. The result is a substantially better prediction than the prior model on both item generalization (0.416 < 0.428) and student generalization (0.468 < 0.473).

Table 3. Knowledge component learning curve model comparison.

	KCs	Recursive grammar skill for 2_step & substitution	Paren skill	Double-ref skill	Item stratified CV (RMSE)	Student stratified CV (RMSE)
No-transfer	3	0	0	0	0.429	0.478
Substitution transfer	3	1	0	0	0.426	0.477
Paren-enhanced	4	1	1	0	0.428	0.473
Double-ref-enhanced	5	1	1	1	0.416	0.468

We have not yet modeled, but have recognized an alternative or additional explanation for the difficulty of the *class* and *jackets* problems. Right expanding forms, which require the "expression => quantity operator expression" rule, may be harder than left expanding forms, which require the "expression => expression operator quantity" rule. This idea garners plausibility from cognitive theory given that right expanding forms may require more cognitive load to maintain the subexpression to be written (e.g., 62-f) while the first part is planned and written (e.g, "62 +"). This analysis predicts that the *trip, class, jackets*, and *rowboat* problems should be more difficult and they are the most difficult 2_step problems.

Future analytic and modeling efforts should pursue these plausible new hidden skills hypotheses and, if confirmed, a close-the-loop study should test whether focused instruction on double reference problems and/or more practice on right expanding expressions yields better learning transfer.

4. SUMMARY AND CONCLUSION

It is worth noting that the control condition in this study is highly similar to the treatment. Many might say, if you practice algebra, you learn algebra. Under that simple analysis, no differences should be expected between the conditions. Further, this control condition is a highly plausible instructional approach supported by a straightforward rational task analysis and by many colleagues who predict it should work: To prepare for story problems involving two operators, practice story problems involving one operator. The detailed data-driven quantitative cognitive task analysis suggested otherwise, in particular, that an inherent difficulty for algebra students learning to symbolize complex story problems is not in the story problem comprehension but in the production of more complex symbolic forms. Isolated practice in producing such forms, as the substitution problems provide, should enhance this hidden cognitive skill and yield better transfer. In a large data pool (711 students) collected in middle school math classes across four school years, our close-the-loop experiment demonstrated strong support for this data-driven prediction.

Our analysis also provides support for cognitive and statistical models that use the same underlying latent constructs (e.g., knowledge components) to predict both task difficulty and task-to-task transfer. This result is not only important to the science of learning, but it has practical relevance to the goal of using data-driven discoveries about domain learning challenges to design instruction for learning transfer. Task

difficulty data can be more easily collected than task-to-task transfer data. Ideal transfer data (i.e., comparing performance on task B when task A is or is not practiced before it) requires giving students curriculum sequences that may harm their learning, therefore, it is costly and ethically challenging. Task difficulty data, when appropriately modeled, provides promise that these cost and ethical challenges can be minimized.

Although this paper does not present new data mining methods, it does indicate that attempts to automatically discover cognitive models, such as LFA [2] and others like it (e.g., Rule Space [22], Knowledge Spaces [24], and matrix factorization [6; 7]) can be used to generate instructional designs that improve student learning and transfer. While innovation in data mining methods is a crucial part of EDM research, it is important to the health of the field and its relevance to society that we pursue more close-the-loop studies and keep the E in EDM!

5. ACKNOWLEDGEMENTS

This work was supported in part by IES award #R305C100024 and NSF award #SBE-0836012.

6. REFERENCES

[1] Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., Evenson, S.E., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J., Beck, J. (2006). Adapting to When Students Game an Intelligent Tutoring System. In *Proc Int Conf Intelligent Tutoring Systems*, 392-401. Jhongli, Taiwan.

[2] Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley, T.-W. Chan (Eds.) *Proc 8th Int Conf ITS*, 164-175.

[3] Clark, R.E., Feldon, D., van Merriënboer, J., Yates, K., & Early, S. (2007). Cognitive task analysis. In J.M. Spector, M.D. Merrill, J.J.G. van Merriënboer, & M.P. Driscoll (Eds.), *Handbook of research on educational communications and technology* (3rd ed., pp. 577–593).

[4] Corbett, A.T. and Anderson, J.R. (1995). Knowledge decomposition and subgoal reification in the ACT programming tutor. In *Proc Artificial Intelligence and Education,1995*. Charlottesville, VA: AACE.

[5] Corbett, A.T., Anderson, J.R., Carver, V.H. and Brancolini, S.A. (1994). Individual differences and predictive validity in student modeling. In A. Ram & K. Eiselt (eds.) In *Proc Sixteenth Annual Conference of the Cog Sci Soc.*

[6] Desmarais MC. (2011). Mapping question items to skills with non-negative matrix factorization. *SIGKDD Explor, 13,* 30–36.

[7] Desmarais M.C. & Naceur, R. (2013). A matrix factorization method for mapping items to skills and for enhancing expert- based Q-matrices. In *Proc Artificial Intelligence and Education*, *2013.* Memphis, TN, 441–450.

[8] Feldon, D. F., Timmerman, B. C., Stowe, K. A., & Showman, R. (2010). Translating expertise into effective instruction: The impacts of cognitive task analysis (CTA) on lab report quality and student retention in the biological sciences. *J Research in Sci Teaching, 47*(10), 1165–1185.

[9] Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy, *Cognitive Science, 7*, 155- 170.

[10] Gick, M. L., & Holyoak, K. J. (1983). Schema induction and analogical transfer. *Cognitive Psychology, 15*, 1-38.

[11] Heffernan, N. & Koedinger, K. R. (1997). The composition effect in symbolizing: The role of symbol production vs. text comprehension. In Shafto, M. G. & Langley, P. (Eds.) *Proc of the 19th Annual Conf Cog Sci Soc*, (pp. 307-312).

[12] Koedinger, K. R. (2002). Toward evidence for instructional design principles: Examples from Cognitive Tutor Math 6. In *Proceedings of PME-NA*, pp. 21-49.

[13] Koedinger, K. R., & Anderson, J. R. (1998). Illustrating principled design: The early evolution of a cognitive tutor for algebra symbolization. *Interactive Learning Environments*, *5*, 161-180.

[14] Koedinger, K.R. & McLaughlin, E.A. (2010). Seeing language learning inside the math: Cognitive analysis yields transfer. In S. Ohlsson & R. Catrambone (Eds.). *Proc 32nd Annual Conf Cog Sci Soc* (pp. 471-476.)

[15] Koedinger, K. R., McLaughlin, E. A., & Stamper, J. C. (2012). Automated Student Model Improvement. In Yacef, K., Zaïane, O., Hershkovitz, H., Yudelson, M., and Stamper, J. (Eds.) *Proc 5th Int Conf on EDM*. (pp. 17-24)

[16] Koedinger, K. R., Stamper, J. C., McLaughlin, E. A., & Nixon, T. (2013). Using data-driven discovery of better student models to improve student learning. In *Proc Int Conf on Artificial Intelligence in Education,* pp 421-430.

[17] Koedinger, K.R., & Tabachneck, H.J.M. (1995). Verbal reasoning as a critical component in early algebra. Paper presented at the annual meeting of the *American Educational Research Association*, San Francisco, CA.

[18] Koedinger, K. R., Yudelson, M., & Pavlik, P.I. (in press). Testing Theories of Transfer Using Error Rate Learning Curves. *Topics in Cognitive Science Special Issue.*

[19] Liu, R., Koedinger, K. R., & McLaughlin, E. A. (2014). Interpreting Model Discovery and Testing Generalization to a New Dataset. In Stamper, J., Pardos, Z., Mavrikis, M., McLaren, B.M. (eds.) *Proc 7th International Conference on Educational Data Mining* (pp.107-113).

[20] Seamster, T.L., Redding, R.E., Cannon, J.R., Ryder, J.M., & Purcell, J.A. (1993). Cognitive task analysis of expertise in air traffic control. *Int J Aviat Psy, 3,* 257–283.

[21] Stamper, J. & Koedinger, K.R. (2011). Human-machine student model discovery and improvement using data. In Biswas, G., Bull, S., Kay, J. & Mitrovic, A. (Eds) *Proc 15th Int Conf, AIED 2011* (pp.353-360).

[22] Tatsuoka KK. (1983).Rule space: an approach for dealing with misconceptions based on item response theory. *J Educ Meas, 20,* 345–354.

[23] Velmahos, G. C., Toutouzas, K. G., Sillin, L. F., Chan, L., Clark, R. E., Theodorou, D., & Maupin, F. (2004). Cognitive task analysis for teaching technical skills in an inanimate surgical skills laboratory. *The American Journal of Surgery, 18*, 114-119

[24] Villano M. (1992). Probabilistic student models: Bayesian belief networks and knowledge space theory. *Proc 2nd Int Conf Intelligent Tutoring Systems,* NewYork: Springer-Verlag.

Does a Peer Recommender Foster Students' Engagement in MOOCs?

Hugues Labarthe
LAMOP
University of Paris I
France (75005)
+33 649487203
hugues.labarthe@ac-creteil.fr

François Bouchet
Sorbonne Universités,
UPMC Univ Paris 06
CNRS, LIP6 UMR 7606
75005 Paris, France
+33 144277135
francois.bouchet@lip6.fr

Rémi Bachelet
Centrale Lille
University of Lille
Lille - France
+33 320335466
remi.bachelet@ec-lille.fr

Kalina Yacef
School of Information Technologies
The University of Sydney
Australia
+61 2 9351 6098
kalina.yacef@sydney.edu.au

ABSTRACT

Overall the social capital of MOOCs is under-exploited. For most students in MOOCs, autonomous learning often means learning alone. Students interested in adding a social dimension to their learning can browse discussion threads, join social medias and may message other students but usually in a blind and somehow random way, only hoping to find someone relevant, available and also willing to interact. This common isolation might be a contributing factor on student attrition rate and on their general learning experience. To foster learners' persistence in MOOCs, we propose to enhance the MOOC experience with a recommender which provides each student with an individual list of rich-potential contacts, created in real-time on the basis of their own profile and activities. This paper describes a controlled study conducted from Sept. to Nov. 2015 during a MOOC on Project Management. A recommender panel was integrated to the experimental users' interface and allowed them to manage contacts, send them an instant message or consult their profile. The population ($N = 8{,}673$) was randomly split into two: a control group, without any recommendations, and an experimental group in which students could choose to activate and use the recommender. After having demonstrated that these populations were similar up to the activation of the recommender, we evaluate the effect of the recommender on the basis of four factors of learners' persistence: attendance, completion, success and participation. Results show the recommender improved all these 4 factors: students were much more likely to persist and engage in the MOOC if they received recommendations than if they did not.

Keywords

Recommender system, MOOC, persistence, social learning.

1. INTRODUCTION

Understanding and reducing the attrition rate in Massive Open Online Courses is still a concern for many scientists, measuring and predicting attrition [2, 10], and trying to uncover its factors [6, 8]. There is a common assumption that students doing well by themselves are more likely to get involved in the learning community. But the paradox is that students do not necessarily know how to initiate and have meaningful conversations within this community, may feel shy or inhibited in such crowded places, which results in further isolation.

Therefore, while learning is above all a social undertaking [1], it turns out that most MOOCs students learn on their own. Far behind the connectivist model, transmissive MOOCs have been implementing functionalities such as synchronous or asynchronous discussions [4], peer grading, potential team mates' geolocation, groups, etc. In such systems, students find others to connect with either in a blind manner or through user-defined filters. Most importantly, contacts are initiated by the students themselves, who need to actively search for others. So it remains extremely difficult to find the right person to interact with in a newly-formed and distance learning MOOC community. This feeling of isolation hinders the learning experience and is a major factor of student attrition [7, 11]. Indeed, the size of students' cohorts and the fact that they usually work at home, at various times and pace, cultivates isolation rather than connection with other students for learning [5], a problem already well-noted before the MOOC era and which led to attempts to reinforce the sense of community [3, 9]. Numerous works have emphasized the need to help people socialising, on the basis that social learning might foster persistence. It requires not only helping students to know how to work with others (and thus to plan tasks for students to perform in a cooperative way), but also in the first place to find relevant potential learning mates one would want to interact with.

In this paper, we address this issue: to foster learners' persistence in MOOCs, we have designed, implemented and tested a recommender system. Our recommender provides each student with a list of high-potential social contacts, on the basis of their own profile and activities. We hypothesise that offering integrated personal data-driven recommendations may increase the students' persistence and success in the MOOC. We chose to consider four key categories of indicators of persistence: attendance, completion, scores and participation.

This paper is organized as follows: in section 2, we present the experiment with our peer recommender, its context and design, the different groups of students considered, the data collected and its preprocessing. In section 3, we analyse the differences in terms of persistence between the experimental groups, and in section 4, we check whether these differences are related to our recommender system. We then conclude the paper with a discussion on limits and on some perspectives of future work.

2. EXPERIMENT WITH A PEER RECOMMENDER

2.1 Context of the experiment

We built a peer recommender system and deployed it during the 6th session of a French Project Management MOOC[1], powered by Unow[2] using a customised version of the Canvas platform [7]. The course lasted 9 weeks, from September to November 2015 and had a total of 24,980 students enrolled. Chronologically, it started with a 4 week long pre-MOOC period (week -3 to -1), where students could perform some self-assessment, introduce themselves on the discussion threads, explore the platform and so on. Then the 4 week-long core part of the MOOC (week 1 to 4 included) took place, with lecture videos, assignments, quizzes and so on. During the remaining 5 weeks (week 5 to 9), students followed their specialisation modules and took their final exam. In parallel to the main MOOC, students could additionally register to two possible streams: (i) an Advanced Certification stream where, in the first four weeks (1 to 4), learners also had to submit three assignments and perform peer-reviews; (ii) a Team track, where students also had to join a team and practice on a real project. The topic of the MOOC being Project Management, this MOOC assumes that learners, in addition to working individually and autonomously to obtain their certification, should also get involved as much as they can in the community. Figure 1 shows the overall MOOC timeline as well as the number of students who reached various checkpoints in the MOOC [e.g. 7716 students took quiz 1 between week 1 (release time) and week 9 (end of the MOOC)].

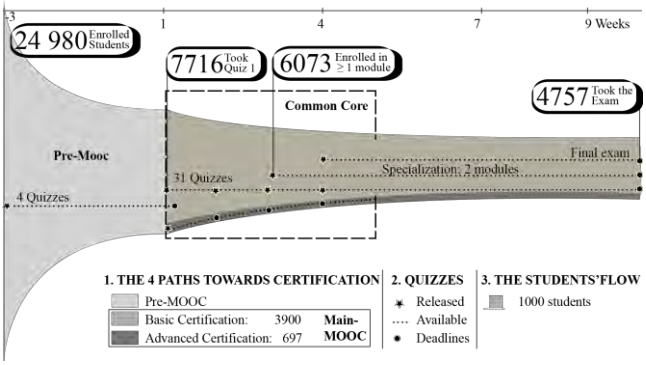


Figure 1. The 6th edition of the Project Management MOOC: a chronological overview

2.2 The peer recommender widget

The recommendation widget is displayed on the navigation bar on the left side of the screen in a space normally empty (cf. Figure 2). It displays 3 lists: a list of suggested contacts in green, a list of contacts marked as favorite in orange and a list of ignored contacts in grey (A). In each list, other students are represented as a thumbnail showing their name and photo (if any). When bringing the mouse pointer over a thumbnail, it also displays the beginning of their biography (if any) as well as 4 icons: one to send a private message, one to contact them through the chat, one to add them as a favorite and one to ignore them (B). The chat widget is shown on the bottom right-hand corner of the interface and minimised by default. When a message is received, an icon is added and a sound played (C). Bringing the mouse pointer over the widget expands it, giving access to two tabs: in the first tab, the favorite contacts appear and a chat can be initiated with up to 6 of them at the same time. The second tab gives access to a list of previous chats, and one can reopen them to keep interacting with the student(s) associated to that chat (D).

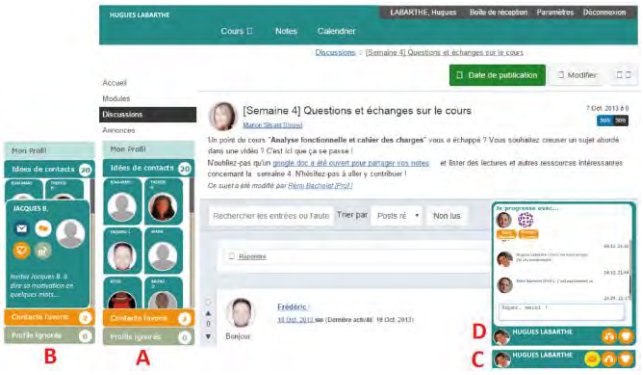


Figure 2. Recommendations and chat widgets

2.3 Experimental Design

In order to evaluate the effect of the recommender system (RS), we performed a controlled study. A set of experimental groups was offered access to the recommender whilst the control group (***Ctl***) was not. Among the experimental groups, some students accepted the use of the recommender (***ToU***) and others did not. Then among those who accepted it, some interacted with it (***Int***) — i.e. managed contacts, consulted profiles and attempted to write messages— and others did not (***No_Int***) — i.e. had the RS widget visible but did not interact at all with it (an interaction being defined as a click on the interface, as mouse-overs were not recorded). The experimental group was also split in three, each subgroup using a different recommendation algorithm (contact suggestions could be either random, based on social features only, or on a combination of social and advancement features). We shall not compare in this paper the efficiency of these algorithms but focus only on the RS' effect.

2.4 Deployment of the Recommender

The recommender was progressively deployed at the beginning of the 4-week core period (week 1 onwards): 100 students on day 1, 4,500 on day 5, 10,000 on day 10. Overall, $N = 8673$ students visiting the platform during this period of time were randomly split between the control group ($N_{Ctl} = 1792$) and the experimental ones ($N_{exp} = 6881$). The experimental group had roughly 3 times more students than the control one because of the aforementioned three subgroups, which will not be considered here. Among students in the experimental groups, $N_{ToU} = 2025$ accepted the recommender Terms of Use (allowing data collection for research purpose) and thus had access to recommendations. Among those students, $N_{Int} = 271$ interacted with the recommendations panel and the chat associated with it (i.e. $N_{No\_Int} = N_{ToU} - N_{Int} = 1754$). Those figures are summarised on Figure 3.

2.5 Data Collection and Pre-processing

We extracted two types of data from the MOOC: learning traces as interaction logs, and demographic information coming from students' answers to a demographic questionnaire they could fill during the Pre-MOOC period, or as they started the MOOC for students arriving late on the platform.

One main way to understand how learners behave is by looking at the interaction logs and the learning records. Overall, 3.95 million

[1] MOOC Project Management, http://mooc.gestiondeprojet.pm/

[2] Unow, http://www.unow.fr/

pages were displayed from Sept. 1st to Nov. 22nd (week -3 to 9) for 373,937 different URLs. We classified them into semantic categories consisting of an action and an area of the website. The URLs combine references to 3 main actions: browsing, viewing content, and downloading resources. Students performed these actions on 12 areas as shown in Table 1. In total, students browsed pages with references to 357 different resources: 8.5% are the homepage, 8.3% lesson pages and 43% quizzes. Many students in developing countries download videos on a third-party website, so these figures should only be used to differentiate students' profiles.

We created 10 variables from this learning dataset to capture students' persistence in the MOOC, which could be grouped into four broad categories: attendance, completion, score and participation. These indicators are shown in Table 2.

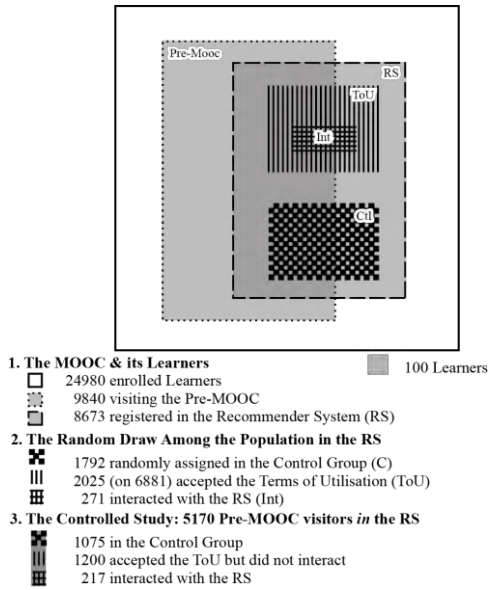


Figure 3. MOOC cohort sizes and overlaps (to scale)

Table 1. Tagging logs towards actions and areas

Categories • subcategories	Brow-sing	View -ing	Down-loading	Total (%)
[homepage]	336,941			8.5
Announcements	27,768			0.7
Assignments	6,602	68,591		1.9
• Syllabus	64,611			1.6
• Corrected assignments		77,270		2.0
• Peer-reviewing materials		59,865		1.5
• Downloaded assignments		69,510	23,606	2.4
Calendar		2,214		0.1
Discussions	35,763	119,777		3.9
Grades	42,961	27,655		1.8
Modules	489,325			12.4
• Badges		80,834		2.0
Others	440			0.0
Pages	7,761			0.2
• Lessons		327,882		8.3
• Other Contents		323,469		8.2
• Downloads	58,981			1.5
Quizzes	11,713	1,686,448		43.0
Profiles		2,678		0.1
TOTAL %	27.4	72.0	0.6	100

Finally, in addition to these learning related variables, we extracted the social features from one of three research surveys filled by participants before Nov. 11th. 10,331 learners completed this survey, from which 1,454 were enrolled in the control group and 5,397 in the experimental groups. 6 variables were considered: student's gender, country, year of birth, their level of study (coded as follow: 0, without A-Level; from 1 to 3: years of university course; 4: master degree; 5: PhD), the previous experiences of MOOCs (0 for newcomers, 2 for experienced with MOOCs; 4 for recurring Project Management MOOC students) and the participation to the Pre-MOOC (0 or 1).

Table 2. Retrieving data related to persistence

Category	Indicators
Attendance	1. Number of days the student visited the platform 2. Number of pages the student accessed 3. Time spent on these pages [max = 600 s]
Comple-tion	4. Number of attempts to complete a quiz 5. Number of quizzes completed
Scores	6. Final score [31 compulsory quizzes + exam]
Participa-tion	7. Number of posts on discussions (forums) 8. Average length of discussion posts 9. Number of messages sent via the Conversations (private messages) 10. Average length of private messages

2.6 Were groups similar before treatment?

In order to assess the similarity between the control group and the experimental ones before the experiment started, we compared their social and behavioral features (cf. Table 3). The data analysis indicates no significant differences between the two groups in terms of gender, countries, year of birth, level of study, previous MOOC experience and attendance on the platform. We can therefore consider the groups were similar before the experiment.

Table 3. Variation between Groups (ANOVA)

Features (number of values)	F	P-value
Gender (2)	0.573348	0.448958
Countries (91)	2.14E-06	0.998834
Year of Birth (59)	3.266974	0.070732
Level of Study (6)	1.195992	0.274163
Previous experiences of MOOCs (3)	0.009721	0.921462
Participation to the Pre-MOOC (2)	0.586452	0.443815

3. GROUP BEHAVIOUR ANALYSIS

Table 4 shows the comparison between 3 groups: the control group (*Ctl*), and among the experimental one, the ones which accepted and did (resp. didn't) use the recommender (*No_Int* - resp. *Int*). Figures show the students who experienced RS were those that displayed the strongest values for the 10 indicators of persistence considered. In particular, the average number of daily visits, pages viewed and duration increase from *Ctl* to *No_Int* and *Int*. The standard deviation increases too, revealing that the highest variation of behavior is observed among those who interacted with the RS. In terms of quizzes, the learners who experienced the RS completed 2 more quizzes than the others and scored on average 17 points higher with a smaller standard deviation. Finally, their participation in discussions and conversations are also higher. Reading these figures, it appears that students who experienced the recommender were also more engaged with the course and its community: even though the 271 students in the *Int* group did not spend so much time online overall, they have managed to obtain higher scores in terms of completion, quiz scores and participation.

However, the fact that students who used the recommender were also more engaged is not sufficient to express causality between the two. The uncertainty resides in the fact that in the experimental group, students could *choose whether or not* to have a recommender widget, and *whether or not to* actually make use of it. It could be the case that, in fact, students who are very engaged are more likely to use the recommender.

Table 4. Average and standard deviation (in italics) of persistence indicators for experimental *versus* control groups

	Attendance from W1 to W4			Completion Nov. 22nd		Scores /100	Participation from W4 to W9			
Indicators	1	2	3	4	5	6	7	8	9	10
Ctl	10	323	1h38	26.4	20	32.2	0.7	69	0.3	31
N=1792	*7*	*285*	*1h57*	*22.5*	*14*	*28.7*	*3.2*	*137*	*2.1*	*127*
No_Int	12	411	2h08	30.5	21.6	36.1	1.4	111	0.6	52
N=1754	*7.5*	*373*	*2h23*	*24*	*13.3*	*30.1*	*5.6*	*190*	*2.1*	*177*
Int	16.1	616	3h46	43.2	26.9	49.1	2.7	154	1.6	107
N=271	*6.9*	*405*	*3h07*	*24.7*	*10*	*27.8*	*6.1*	*186*	*3.8*	*212*

4. EFFECTS OF THE RECOMMENDER

To determine the RS' real effect on learners' persistence, we need to compare cohorts that were similar in terms of persistence before the experiment started and see how they evolve during the course of the MOOC. For example, we want to find out whether, among students who were very passive before the recommender was made available, a larger proportion of those who used the recommender persisted in the MOOC. To do so, we first clustered students during the Pre-MOOC period (i.e. before they were allocated to a group, and before the RS was made available) based on their level of engagement (section 4.1). We then, in each cluster, analysed the control and experimental groups according to each dimension of persistence at the end of the main MOOC period.

4.1 Pre-MOOC activity clusters

To cluster students in the Pre-MOOC period, we used as features the times spent on 18 of the actions in areas shown in Table 1 (i.e. excluding those related to material not yet available). During the Pre-MOOC, 294,209 pages were accessed by the 9,840 students who were present in the Pre-MOOC period. We used the k-means algorithm to extract clusters and found the best solution involved 4 groups, shown in Table 5 and called A, B, C D on the basis of their time spent (A being the most active and D the least). Students in cluster A spent over 1h40 on the website viewing lessons, quizzes and discussions (sum of the mean values). The second cluster (B) spent less than 40 minutes, essentially in the quizzes area; in the third cluster, C, the time is even shorter and those in the last one, D, stayed less than 2 min on the website in total.

Table 6 shows the distribution across the 4 Pre-MOOC clusters of students who would later belong to groups *Ctl*, *No_Int* and *Int*. Since we want to follow the evolution of the students who were present in the Pre-MOOC period, we must only consider the intersecting population. The populations of the various groups are now: $N_{Pre\&Int} = 217$ students who interacted with the recommender (vs. $N_{int} = 271$); $N_{Pre\&No\_Int} = 1{,}200$ (vs. $N_{No\_Int} = 1{,}754$) who accepted its ToU without using it; $N_{Pre\&Ctl} = 1{,}075$ (vs. $N_{Ctl} = 1{,}792$) who were randomly enrolled in the control group.

To deal with the sample size difference and compare the features of students in Int with students in *Ctl* and *No_Int*, a subsample was ten times randomly drawn for each cluster – e.g. in the PreMOOC_D cluster, 77 persons out of 551 were ten times randomly drawn. The percentage averages in tables 8, 10 and 12 are computed only on the basis of features of students from these subsamples. We will now exclusively focus on the last 3 Pre-MOOC clusters since the most active group (PreMOOC_A) is very small (8) and already very engaged.

Table 5. Interactions and clusters during the Pre-MOOC

Features (in seconds)	PreMooc_D	PreMooc_C	PreMooc_B	PreMooc_A
browsing_homepage	21	48	149	411
browsing_announcements	1	4	15	81
browsing_assignment	4	14	48	210
browsing_discuss._topics	2	8	26	190
browsing_grades	1	3	11	30
browsing_modules	7	43	140	428
browsing_pages	0	1	6	8
browsing_quizzes	0	1	2	2
downloading_assignment	0	0	0	2
viewing_assignment	1	11	49	208
viewing_calendar_events	0	0	0	7
viewing_discuss._topics	13	82	226	857
viewing_grades	0	0	1	1
viewing_modules	0	7	24	65
viewing_pages	25	163	550	1472
viewing_profiles	0	1	2	37
viewing_quizzes	33	768	1167	1965

Table 6. Clusters and Groups during the Pre-MOOC

	N (%)	N	Ctl	No_Int	Int
PreMooc_D	66	6,386	551	578	77
PreMooc_C	26	2,534	393	404	78
PreMooc_B	7	658	118	190	54
PreMooc_A	1	62	13	28	8
Total	100	9,640	1,075	1,200	217

4.2 Attendance during the Common Core

We clustered all enrolled students (N=24,980) using the full set of features in Table 4 for a total of 3,110,321 pages seen during the Common Core. We obtained 4 clusters, shown in Table 7, named according to their attendance quality (A the best, D the worst). Cluster Att_D, with 77% students, has the poorest overall mean in regards to all the features, not exceeding 6 minutes spent interacting with all pages. The mean values of the second cluster, Att_C (with 17% students), total around 1h30min. The two last clusters, Att_B and Att_A, contain 3% each of the population: the main difference is the time spent by Att_A in the assignments area.

We then explored how the pre-MOOC students evolve into these attendance clusters, according to their activities during the Common Core (cf. Table 8, where figures in a row represent 100% of the mentioned *Ctl*, *No_Int* and *Int*). Considering the lower clusters D to B, these figures suggest that the recommender system played a significant role on the duration of the visits of the learners from clusters D, C and B, that is to say 99% of the Pre-MOOC population. Indeed, one can see that students who used to be in D, having the RS marginally increased their persistence, but significantly increased the persistence of students who used it (32% of them now being in cluster B vs. 8% for students of the control group). For students in clusters C and B during the pre-MOOC, we observe a similar pattern: simply having access to the RS tended to increase their persistence, and actually using the RS tended to significantly decrease their chance of dropping out (i.e. ending up in cluster D, the least active students).

Table 7. Interactions and clusters during the Common Core

Features (in seconds)	At_D	At_C	At_B	At_A
others	0	0	1	2
browsing_	15	214	554	856
browsing_announcements	1	12	53	61
browsing_assignments	5	48	181	155
browsing_discussion_topics	2	23	90	315
browsing_grades	1	32	160	276
browsing_modules	22	430	1022	1249
browsing_pages	0	4	4	6
browsing_quizzes	0	7	7	6
downloading_assignments	0	3	5	144
viewing_assignments	7	248	636	9334
viewing_calendar_events	0	1	11	5
viewing_discussion_topics	14	127	467	1477
viewing_grades	0	10	48	216
viewing_modules	3	57	169	177
viewing_pages	67	1025	2766	2398
viewing_profiles	0	1	4	18
viewing_quizzes	180	3257	8286	5165
% students	77	17	3	3

Table 8. Attendance: Evolution of the learners from the Pre-MOOC to the Core-MOOC periods

↓From To→	At_D	At_C	At_B	At_A	Group
PreMooc_D 66%	39	49	8	4	Ctl
	33	49	12	7	No_Int
	9	39	32	19	Int
PreMooc_C 26%	26	50	9	16	Ctl
	24	43	12	20	No_Int
	17	45	12	27	Int
PreMooc_B 7%	16	48	12	24	Ctl
	16	38	15	31	No_Int
	2	37	20	41	Int

4.3 Completion and final scores

We clustered again the student population, using scores and activity in the examination points (i.e. scores obtained at the 31 quizzes and the final exam by the end of the MOOC). Each score is standardised to marks out of 100. We obtained again 4 clusters, which centroids are shown in Table 9. The values of the centroid of the first cluster indicates a large part of students (71%) who participated in the first 2 quizzes but obtained a very low score on them and then did not participate again in any assessment. The centroid of the second cluster (4% of learners) corresponds to students who easily passed the quizzes of the first week but dropped out on the second. The third cluster (4%) has similar students, but who gave up in week 3. Finally, the last cluster (21%) contains all the students who completed all the quizzes and final exam with high scores in each.

Once again figures in Table 10 show that, by accepting the recommendations and, even more, interacting with its panel, the learners went closer to completion and obtained better scores. In particular, we observe as before for students in clusters D and B that the mere presence of the RS has a small positive impact on their chances to complete (or at least to stay longer on the MOOC before giving up), but that students who use the RS benefit the most from an increased chance to complete. For students in cluster C, the use of the RS seems to have made some of them drop out overall a bit later (week 2 instead of week 1) but did not increase their chance to complete the MOOC.

Table 9. Completion and score clusters during whole MOOC

Week	Quiz	D	C	B	A	Week	Quiz	D	C	B	A
1	1	3	92	92	96	2	17	0	1	67	92
	2	1	82	82	87		18	0	0	48	83
	3	0	92	92	96		19	0	1	57	95
	4	0	82	89	95	3	20	0	1	39	92
	5	0	76	93	98		21	0	1	40	96
	6	0	54	78	87		22	0	1	36	95
	7	0	63	92	98		23	0	1	33	91
2	8	0	26	93	96		24	0	1	31	94
	9	0	18	94	97		25	0	1	29	89
	10	0	10	92	95	4	26	0	1	10	91
	11	0	7	88	93		27	0	1	5	93
	12	0	4	85	93		28	0	0	2	90
	13	0	2	83	93		29	0	1	1	96
	14	0	2	86	95		30	0	0	1	95
	15	0	1	76	89		31	0	0	1	86
	16	0	1	75	93		EXAM	0	1	3	78
N (%)		71	4	4	21	N (%)		71	4	4	21

Table 10. Completion and final scores: Evolution of the learners from the Pre-MOOC to the Core-MOOC periods

↓From To→	Co_D	Co_C	Co_B	Co_A	Group
PreMooc_D 66%	32	5	13	49	Ctl
	27	6	14	53	No_Int
	10	5	4	81	Int
PreMooc_C 26%	15	9	11	65	Ctl
	9	9	14	69	No_Int
	8	14	13	65	Int
PreMooc_B 7%	8	5	8	79	Ctl
	5	9	14	73	No_Int
	4	2	11	83	Int

4.4 Participation to the Common Core

The total number and average length of the messages sent by each student were retrieved from the Canvas database (discussions and conversations). Using k-means with features from the participation section of Table 2, we obtained once again 4 clusters, shown in Table 11: a first cluster, Pa-D (89% of 24,980 enrolled learners) did not interact at all with others. The centroid of the second one indicates 2 posts of an average of 237 characters on the discussion topics (9%). The third cluster (2%) seems to have a similar activity but slightly stronger in term of number of posts (2.7) and average post length (599 characters). The last 1% is highly committed to the course and its community: most of them correspond to students who were part of the advanced certification stream.

Table 12 shows how students in the Pre-MOOC clusters are distributed over the 4 participation clusters at the end of the MOOC. Figures reveal a consistent positive effect of the mere presence of the RS across the initial Pre-MOOC clusters: there are always less students in cluster Pa_D in the *No_Int* group than in the control group. Less surprisingly, students who interacted with the RS generally did so to send a message to someone, so they overall also ended up less often being in a situation where they do not interact at all with anyone else (complete isolation). Finally, we can see that merely giving students access to a recommender panel does not prevent them from being social-lazy: a majority (82%, 88% 69% respectively in clusters D, C and B) of the students who interacted with the RS did not attempt to directly contact anyone else. These figures are however probably lower than they would be if every student had access to the associated direct chat module, and still better than in the Control group (96%, 91% and 80% respectively

in clusters D, C and B) who could only contact others in a blind way through the forum or private messages.

Table 11. Participation Clusters of all enrolled students

Attribute	Pa-D	Pa-C	Pa-B	Pa-A
Nb** of discussions	0	2	2	9
Discussions length*	2	237	599	264
Nb** of conversations	0	0	0	7
Conversations length*	1	9	19	542
N%	89	9	2	1

*: average number of characters; **: number of posts/messages sent

Table 12. Participation: Evolution of the learners from the Pre-MOOC to the Core-MOOC periods

↓From To→	Pa_D	Pa_C	Pa_B	Pa_A	Group
PreMooc_D 66%	78	18	2	2	Ctl
	67	25	4	4	No_Int
	47	35	6	12	Int
PreMooc_C 26%	76	15	4	5	Ctl
	69	18	4	9	No_Int
	62	26	4	9	Int
PreMooc_B 7%	66	14	4	15	Ctl
	53	25	6	15	No_Int
	39	30	7	24	Int

5. Discussion, conclusion and perspectives

We conducted a controlled study during a Project Management MOOC, in which a recommender panel integrated to the user interface provided suggestions and allowed contact management, instant messaging and profile consultation. Students were randomly split into a control group (without any recommendations), and an experimental group (in which they could activate and use our recommender). The number of the students involved in this experience was relatively high: among 6881 selected students, 2025 accepted the Term of Use of the recommender and 279 accessed its functionalities. We have shown that these populations were similar before the activation of the recommender, and evaluated its effect according to four categories of indicators relative to learners' persistence: attendance, completion, success and participation. Results suggested that our recommender improved these four categories of indicators: students are much more likely to persist and engage in the MOOC if they receive recommendations than if they do not.

The main interest was then to evaluate the effect the recommendations might have played in such increased rates of engagement. To do so, we focused on clustering similar learners according to their activities before the beginning of the course, leading to four groups from the least (D) to the most (A) active students. We analysed the way 3 of these 4 groups (representing 99% of the students) were evolving in terms of attendance, completion and score, participation. We observed overall a significant improvement of students' engagement, not only for those who interacted with the recommendations, but, more largely, for all of those accepted using the recommendation system.

This study presented several limitations: (1) for experimental purposes, we restricted the access to the direct communication tool; (2) since not all students had access to the RS and the chat, the teaching team could not use them for pedagogical activities, which could have boosted the effect of the RS; (3) students in the control group were not asked to accept the RS Terms of Use, since they would not be given access to it – however, while it is thus possible that students who accepted the ToU were more motivated, the analysis presented in section 2.6 shows that students in the control and experimental groups were similar in terms of participation before the beginning of the core MOOC and demographics.. Furthermore, the most significant results were obtained comparing students who interacted vs. those who did not interact with the RS, and these results are not affected.

Overall, this controlled study is highly supporting the idea that recommending learners to learners, in such crowded places as MOOC platforms, is an effective way to get them more involved in terms of attendance, completion, scores and participation. In the future, we intend to look into more details the impact of the different recommendation strategies, and the different ways students interacted with the recommendation system.

6. ACKNOWLEDGMENTS

This work was funded by the French Educational Board and by the Human-Centred Technology Cluster of the University of Sydney. We thank Unow for letting us deploy our RS on their MOOC.

7. REFERENCES

[1] Bandura, A. 1971. *Social Learning Theory*. General Learning Corporation.

[2] Bouchet, F. and Bachelet, R. 2015. Do MOOC students come back for more? Recurring Students in the GdP MOOC. *Proc. of the European MOOCs Stakeholders Summit 2015* (Mons, Belgium), 174–182.

[3] Croft, N., Dalton, A. and Grant, M. 2010. Overcoming Isolation in Distance Learning: Building a Learning Community through Time and Space. *Journal for Education in the Built Environment*. 5, 1, 27–64.

[4] Ferschke, O., Yang, D., Tomar, G. and Rosé, C.P. 2015. Positive Impact of Collaborative Chat Participation in an edX MOOC. *Proc. of Artificial Intelligence in Education: 17th International Conference, AIED 2015, Madrid, Spain, June, 2015.* Springer. 115–124.

[5] Gütl, C., Rizzardini, R.H., Chang, V. and Morales, M. 2014. Attrition in MOOC: Lessons Learned from Drop-Out Students. *Proc. of Learning Technology for Education in Cloud. MOOC and Big Data*. Springer. 37–48.

[6] Kizilcec, R.F., Piech, C. and Schneider, E. 2013. Deconstructing Disengagement: Analyzing Learner Subpopulations in Massive Open Online Courses. *Proc. of the Third International Conference on Learning Analytics and Knowledge* (New York, NY, USA), 170–179.

[7] Labarthe, H., Bachelet, R., Bouchet, F. and Yacef, K. 2016. Towards increasing completion rates through social interactions with a recommending system. *Proc. of the European MOOCs Stakeholders Summit 2016* (Graz, Austria), 471-480.

[8] Rosé, C.P., Carlson, R., Yang, D., Wen, M., Resnick, L., Goldman, P. and Sherer, J. 2014. Social Factors That Contribute to Attrition in MOOCs. *Proc. of the First ACM Conference on Learning @ Scale Conference* (New York, NY), 197–198.

[9] Rovai, A.P. 2002. Building Sense of Community at a Distance. *The International Review of Research in Open and Distributed Learning*. 3, 1.

[10] Yang, D., Sinha, T., Adamson, D. and Rosé, C.P. 2013. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. *Proc. of the NIPS Data-Driven Education Workshop*.

[11] Yang, D., Wen, M. and Rosé, C.P. 2014. Peer Influence on Attrition in Massive Open Online Courses. *Proc. of the 7th International Conference on Educational Data Mining* (London, UK), 405–406.

A Contextual Bandits Framework for Personalized Learning Action Selection

Andrew S. Lan
Rice University
mr.lan@sparfa.com

Richard G. Baraniuk
Rice University
richb@sparfa.com

ABSTRACT

Recent developments in machine learning have the potential to revolutionize education by providing an optimized, personalized learning experience for each student. We study the problem of selecting the best personalized learning action that each student should take next given their learning history; possible actions could include reading a textbook section, watching a lecture video, interacting with a simulation or lab, solving a practice question, and so on. We first estimate each student's knowledge profile from their binary-valued graded responses to questions in their previous assessments using the SPARFA framework. We then employ these knowledge profiles as contexts in the contextual (multi-armed) bandits framework to learn a policy that selects the personalized learning actions that maximize each student's immediate success, i.e., their performance on their next assessment. We develop two algorithms for personalized learning action selection. While one is mainly of theoretical interest, we experimentally validate the other using a real-world educational dataset. Our experimental results demonstrate that our approach achieves superior or comparable performance as compared to existing algorithms in terms of maximizing the students' immediate success.

1. INTRODUCTION

In traditional classrooms, learning has largely remained a "one-size-fits-all" experience in which the instructor selects a single learning action for all students in their class, regardless of their diversity in backgrounds, learning goals, and abilities. The quest for a fully personalized learning experience began with the development of intelligent tutoring systems (ITSs) [6, 19, 38, 40]. However, to date, ITSs are primarily *rules-based*, meaning that building an ITS requires domain experts to consider every possible learning scenario that students can encounter and then manually specify the corresponding learning actions in each case. This approach is not scalable, since it is both labor-intensive and domain-specific.

Machine learning-based personalized learning systems [30] have shown great promise in reaching beyond ITS to scale to large numbers of subjects and students. These systems automatically create *personalized learning schedules*, a series of *personalized learning actions* (PLAs) for each individual student to take that maximizes their learning. Examples of PLAs include reading a textbook section, watching a lecture video, interacting with a simulation or lab, solving a practice question, etc. Instead of domain-specific rules, machine learning algorithms are used to select PLAs automatically by analyzing the data students generate as they interact with learning resources.

The general problem of creating a fully personalized learning schedule for each student can be formulated using the partially observed Markov decision process (POMDP) framework [31]. POMDPs utilize models on the students' latent knowledge states [23, 28] and their transitions [8, 11, 18, 22] to learn a PLA selection policy (a mapping from the knowledge state space to the set of learning actions) that maximizes a reward received in the possibly distant future (long-term learning outcome). Previous work applying POMDPs to personalized learning have achieved some degree of success [4, 9, 32, 33]. However, learning a personalized learning schedule using a POMDP is greatly complicated by the curse of dimensionality; the solution quickly becomes intractable as the dimensions of the state and action spaces grow [31]. Consequently, POMDPs have made only a limited impact in large-scale personalized learning applications involving large numbers of students and learning actions.

A more scalable approach to personalized learning is to learn a PLA selection policy using the *multi-armed bandits* (MAB) framework [10, 27], which is more suitable to optimizing students' success on immediate follow-up assessments (short-term learning outcome). The simplicity of the MAB framework makes it more practical than the POMDP framework in real-world educational applications, since it requires far less training data.

1.1 Contributions

In this paper, we study the problem of selecting PLAs for each student given their learning history using MABs. We first estimate each student's latent concept knowledge profile from their learning history (specifically, their binary-valued graded responses to questions in previous assessments) using the sparse factor analysis (SPARFA) framework [23]. Then, we use these concept knowledge profiles as contexts in the contextual (multi-armed) bandits framework to learn a policy to select PLAs for each student that maximize their performance on the follow-up assessment.

We develop two algorithms for PLA selection. The first algorithm, CLUB, has theoretical guarantees on its ability to identify the optimal PLA for each student. The second algorithm, A-CLUB, is more intuitive and practical; we experimentally validate its performance using a real-world educational dataset. Our experimental results demonstrate

that A-CLUB achieves superior or comparable performance to existing algorithms in terms of maximizing students' immediate success.

1.2 Related work

The work in [27] applies an MAB algorithm to educational games in order to trade off scientific discovery (learning about the effect of each learning resource) and student learning. Their approach is context-free and thus not ideally suited for applications with significant variation among the knowledge states of individual students. Indeed, it can be seen as a special case of our work in this paper when there is no context information available.

The work in [36] applies a contextual bandits algorithm to the problem of selecting the optimal PLA for each student given their previous exposure to learning resources. In their approach, each dimension of the context vector corresponds to the students' exposure to one learning resource. Thus, the context space quickly grows large as the number of learning resources increases. Our approach, in contrast, performs dimensionality reduction on student learning histories using the SPARFA framework and uses the resulting student concept knowledge profiles as contexts. This feature enables our approach to be applied to datasets where student learning histories contain a large number of learning resources.

The work in [29] collects high-dimensional student–computer interaction features as they play an educational game and uses them to search for a good teaching policy. We emphasize that our approach can be applied to almost all educational applications, not just computerized educational games, since it only requires graded response data of some kind.

The works in [10] and [20] both use some form of expert knowledge to learn a teaching policy. The approach of [10], in particular, uses expert knowledge to narrow down the set of possible PLAs a student can take. Our approach, in contrast, requires no expert knowledge and is therefore fully data-driven and domain-agnostic.

The work in [26] fuses MAB algorithms with Gaussian process regression in order to reduce the amount of training data required to search for a good teaching policy. Their work requires the policy to be parameterized by a few parameters, while our framework does not and can thus learn more complicated policies using only reward observations.

The work in [35] found that various student response models, including knowledge tracing (KT) [11], IRT models [28, 34, 5], additive factor models (AFM) [8], and performance factor models (PFM) [16] can have similar predictive performance yet lead to very different teaching policies. While these results are indeed interesting, we emphasize that the focus of the current work is to develop policy learning algorithms rather than comparing student models.

2. PROBLEM FORMULATION

We study the problem of creating a personalized learning schedule for each student by selecting the PLA they should take based on their prior learning experience. We assume that a student's learning schedule consists of a series of assessments with PLAs embedded in between, a setting that is typical in traditional classrooms, blended learning environments, and online courses like MOOCs [12, 13]. Each PLA can correspond to studying a learning resource, e.g., reading a textbook section, watching a lecture video, conducting an interactive simulation, solving a practice question, etc., or a combination of several learning resources.[1] Assessment could be a pop-quiz with a single question, a homework set with multiple questions, or a longer exam. Each student's personalized learning schedule can be visualized as in Figure 1, where a PLA is taken between consecutive assessments (starting after Assessment 1).

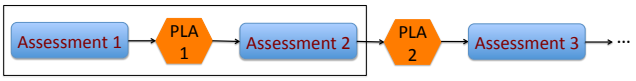


Figure 1: A personalized learning schedule.

The goal of this work is to select the optimal PLA for each student given their learning history (their graded responses to previous assessments) that maximizes their immediate success, i.e., the credit they receive on the following assessment. We aim to learn this learning action selection rule from data. For simplicity of exposition, we will place PLA 1 between Assessment 1 and Assessment 2 (as encased in the box in Figure 1) as a running example throughout the paper.

Let A denote the total number of PLAs available, let K denote the number of latent concepts covered up to Assessment 1, and let Q denote the number of questions in Assessment 2, with $s_i, i = 1, \ldots, Q$ the maximum credit of each question. Let $Y_{i,j}$ denote the binary-valued graded response of student j to question i, with $Y_{i,j} = 1$ denoting a correct response and $Y_{i,j} = 0$ an incorrect response. In order to pin down a feasible PLA selection algorithm, we make two simplifying assumptions: i) We assume that a reliable estimate of each student's latent concept knowledge vector (estimated from their graded responses to Assessment 1), denoted by $\mathbf{c}_j \in \mathbb{R}^K$, is available to the PLA selection algorithm. Such an estimate can be obtained using any IRT-like method, e.g., SPARFA [23]. ii) We assume that the PLA selected for each student will directly affect their performance on Assessment 2.

With this notation in place, we can restate our goal as selecting a PLA for student j, given their current concept knowledge[2] $\mathbf{c}_j$ in order to maximize their performance (i.e., their expected credit $\sum_{i=1}^{Q} s_i \mathbb{E}[Y_{i,j}]$) on Assessment 2.

2.1 Background on bandits

The multi-armed bandit (MAB) framework [3] studies the problem of a player trying to learn a policy that maximizes the total expected reward by playing (pulling the arms of) a collection of slot machines with a fixed number of trials and no prior information about each machine. Each machine has a fixed reward distribution that is unknown to the player. The key to maximizing the total expected reward is to find the right balance between exploration (playing

[1] Our notion of PLA is very general, and we do not restrict ourselves to studying a single learning resource.

[2] In practice, we augment $\mathbf{c}_j$ as $[\mathbf{c}_j^T \; 1]^T$ to add an "offset" parameter to each arm.

machines that might yield high rewards) and exploitation (repeatedly playing the machine with the highest observed reward). Analogously, a personalized learning system must strike a balance between testing the efficacy of every learning action (exploration) and maximizing the students' learning outcomes using observations on the actions (exploitation) [27].

Contextual (multi-armed) bandits [1, 2, 15, 24, 37] extend the MAB framework by accounting for the existence of additional information on the player and/or the machines, referred to as "contexts", in order to improve the policy. Our PLA selection problem fits squarely the contextual bandits framework, where the current estimates of students' concept knowledge correspond to the contexts and each PLA corresponds to an arm. Pulling an arm corresponds simply to selecting a PLA. In this paper, the context will include only information on the students. See Sec. 5 for a discussion on extending our framework to incorporate information on the learning resources into the contexts.

3. ALGORITHMS

The two algorithms we develop in this section are so-called upper confidence bound (UCB)-based algorithms [3]. These algorithms maintain estimates of the expected reward of each arm together with confidence intervals around these estimates, and iteratively update them as each new pull and its corresponding reward is observed. They then pull the arm with the highest UCB on the reward, which is equal to the expected reward plus the width of the confidence interval.

3.1 CLUB: An algorithm in theory

We first develop the *contextual logistic upper confidence bound* (CLUB) algorithm in order to provide theoretical guarantees for the PLA selection problem. We assume that the binary-valued student responses to the questions in Assessment 2 are Bernoulli random variables with success probabilities following a logistic model

$$p(Y_{i,j_{a_s}}=1)=\Phi_{\log}(\mathbf{c}_{j_{a_s}}^T\mathbf{w}_i^a)=\frac{1}{1+e^{-\mathbf{c}_{j_{a_s}}^T\mathbf{w}_i^a}},\ s=1,\ldots,n_a,$$

where $\mathbf{w}_i^a \in \mathbb{R}^K$ is the parameter vector that characterizes the students' responses to question i after taking PLA a. Also, j_{a_s} denotes the index of the s^{th} student taking PLA a, and n_a denotes the total number of students taking PLA a. $\Phi_{\log}(\cdot)$ denotes the inverse logit link function.

The maximum-likelihood estimate (MLE) of $\mathbf{w}_i^a$ is

$$\widehat{\mathbf{w}}_i^a = \arg\min_{\mathbf{w}} -\sum_{s=1}^{n_a}\log p(Y_{i,j_{a_s}}|\mathbf{c}_{j_{a_s}},\mathbf{w}), \qquad (1)$$

which can be computed using standard logistic regression algorithms [17] whenever the MLE exists (see [39, Sec. 5.1] for a detailed discussion on the conditions under which the MLE exists).

As detailed in Algorithm 1, CLUB maintains MLEs of the parameter vector $\mathbf{w}_i^a$ of each PLA together with a confidence interval around it. Then, after receiving a student's concept knowledge vector $\mathbf{c}_j$, CLUB selects the PLA with the highest UCB on the expected credit on the student's following assessment.

Algorithm 1: CLUB

Input: A set of student concept knowledge state estimates $\mathbf{c}_j$, $j=1,2,\ldots$, and parameters $\lambda_0,\delta,\eta,\epsilon$
Output: PLA a_j for each student, $j=1,2,\ldots$

```
MLE_all exist ← False, n_a ← 0, ∀a
for j ← 1 to ∞ do
    if MLE_all exist then
        Estimate ŵ_i^a, ∀i,a according to (1)
        Σ_a ← λ_0 I_K + Σ_{s=1}^{n_a} c_{j_as} c_{j_as}^T, ∀a
        a_j ←
        arg max_a Σ_{i=1}^{Q} s_i(Φ_log(c_j^T ŵ_i^a) + c_i(n_a) sqrt(c_j^T Σ_a^{-1} c_j))
    else
        Randomly select a_j among PLAs where ∃i s.t. ŵ_i^a
        does not exist
    n_{a_j} ← n_{a_j} + 1
    MLE_all exist ← True
    for a ← 1 to A do
        for i ← 1 to Q do
            if ŵ_i^a does not exist (verified via [39, Thm. 2])
            then
                MLE_all exist ← False
```

The constants in Algorithm 1 are given by $c_i(n_a) = \sqrt{2K(3+2\log(1+2a_m^2/\lambda_0))\log n_a K/\delta}/b_{i,a}$, where $a_m = \sqrt{K+2\sqrt{K\log(1/\eta)}+2\log(1/\eta)}$ and $b_{i,a} = 1/(2+e^{\|\mathbf{w}_i^a\|_2 a_m}+e^{-\|\mathbf{w}_i^a\|_2 a_m})$, and $0<\delta,\eta\ll 1$. Algorithm 1 exhibits theoretical optimality guarantees (omitted due to space constraints and available at `www.sparfa.com` [21]).

3.2 A-CLUB: An algorithm in practice

Since in practice we do not know the values of the constants $\Delta_{a,j}$ and also need to set the parameters ϵ, δ, and η, Algorithm 1 and its theoretical guarantees are not directly applicable. Furthermore, as the number of students grows, the confidence bounds around the estimates of each PLA's parameters might become overly pessimistic, causing the algorithm to over-explore [15]. Therefore, we now develop a second CLUB-like algorithm that leverages the asymptotic normality of the MLEs of the PLA parameters [14]. The asymptotic normality property states that, as the number of students grows large, the estimation error of the parameter $\mathbf{w}_i^a$ for each PLA converges to a normally distributed random vector with zero mean and a covariance matrix that is a scaled inverse of the Fisher information matrix

$$\mathbf{F}_a := \sum_{s=1}^{n_a}\frac{\mathbf{c}_{j_{a_s}}\mathbf{c}_{j_{a_s}}^T}{2+e^{\mathbf{c}_{j_{a_s}}^T\mathbf{w}_i^a}+e^{-\mathbf{c}_{j_{a_s}}^T\mathbf{w}_i^a}}.$$

Thus, we can build a confidence ellipsoid around the point estimate generated by (1), albeit asymptotically. In practice, since the true values of the parameters $\mathbf{w}_i^a\ \forall i,a$ are unknown, we will use their estimates $\widehat{\mathbf{w}}_i^a$ to approximate the Fisher information matrix.

Armed with the confidence ellipsoid, we can now compute the upper bound of the expected response of student j on each question in Assessment 2 after taking PLA a. This cor-

Algorithm 2: A-CLUB

Input: A set of student concept knowledge state estimates, $\mathbf{c}_j$, $j = 1, 2, \ldots$, parameter α
Output: PLA a_j for each student
$\text{MLE}_{\text{all exist}} \leftarrow$ False, $n_a \leftarrow 0$, $\forall a$
for $j \leftarrow 1$ **to** ∞ **do**
  if $\text{MLE}_{\text{all exist}}$ **then**
    Estimate $\widehat{\mathbf{w}}_i^a$, $\forall i, a$ according to (1)
    $\mathbf{F}_a \leftarrow \lambda_0 \mathbf{I}_K + \sum_{s=1}^{n_a} \frac{\mathbf{c}_j \mathbf{c}_j^T}{2+e^{\mathbf{c}_{j_s}^T \mathbf{w}_i^a} + e^{-\mathbf{c}_{j_s}^T \mathbf{w}_i^a}}$, $\forall a$
    $a_j \leftarrow$
    $\arg\max_a \sum_{i=1}^{Q} s_i \Phi_{\log}(\mathbf{c}_j^T \widehat{\mathbf{w}}_i^a + \sqrt{\alpha(\mathbf{c}_j^T \mathbf{F}_a^{-1} \mathbf{c}_j)/n_a})$
  else
    Randomly select a_j among PLAs where $\exists i$ s.t. MLE of $\mathbf{w}_i^a$ does not exist
  $n_{a_j} \leftarrow n_{a_j} + 1$
  $\text{MLE}_{\text{all exist}} \leftarrow$ True
  for $a \leftarrow 1$ **to** A **do**
    for $i \leftarrow 1$ **to** Q **do**
      if MLE does not exist for $\mathbf{w}_i^a$ (verified via [39, Thm. 2]) **then**
        $\text{MLE}_{\text{all exist}} \leftarrow$ False

responds to the following constrained optimization problem[3]

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & -\frac{1}{1+e^{-\mathbf{c}_j^T \mathbf{w}}} \\ \text{subject to} \quad & (\mathbf{w} - \widehat{\mathbf{w}}_i^a)^T \mathbf{F}_a (\mathbf{w} - \widehat{\mathbf{w}}_i^a) \leq \alpha / n_a, \end{aligned}$$

where α is a parameter controlling the size of the confidence ellipsoid and thus the amount of exploration. The solution to this problem is given by $\mathbf{w} = \widehat{\mathbf{w}}_i^a + \sqrt{\frac{\alpha}{n_a \mathbf{c}_j^T \mathbf{F}_a^{-1} \mathbf{c}_j}} \mathbf{F}_a^{-1} \mathbf{c}_j$. Therefore, we obtain an upper bound for the expected grade for student j on question i after taking PLA a as $\Phi_{\log}(\mathbf{c}_j^T \widehat{\mathbf{w}}_i^a + \sqrt{\alpha \mathbf{c}_j^T \mathbf{F}_a^{-1} \mathbf{c}_j / n_a})$. We thus arrive at Algorithm 2, which we dub asymptotic CLUB (A-CLUB).

4. EXPERIMENTS

In this section, we validate our algorithms experimentally on personalized cohort selection using a college physics course dataset. We will compare the performance of Algorithm 2 against other baseline (contextual) MAB algorithms. We do not compare Algorithm 1, since its theoretical bounds are usually too pessimistic in practice [15]. For comparisons using two additional datasets, see [21].

Dataset. The dataset consists of the binary-valued graded responses in a semester-long physics course administered on OpenStax Tutor [30] with $N = 39$ students answering 286 questions. Cognitive science experiments were conducted in this course to test the effect of spacing versus massed practice on the students' long-term retrieval performance of knowledge [7]. For this purpose, the students were randomly divided into two cohorts containing 19 and 20 students. There are

[3]We assume $\mathbf{c}_j$ is non-zero; otherwise we would simply select a PLA at random.

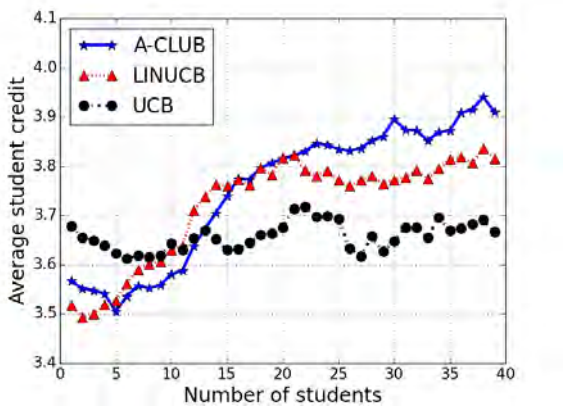


Figure 2: Average student credit on Assessment 5 vs. number of students used by three algorithms. Student performance on the follow-up assessment increases as the algorithms have access to more training data. Concretely, using data from 38 students, A-CLUB finds a PLA selection policy whereby students perform approximately 10% better than selecting randomly.

a total of 11 weekly assessments and 3 review assessments throughout the course. In the first three assessments, both cohorts received the same set of assessment questions. Starting from Assessment 4, apart from the same set of assessment questions both cohorts received on the concepts covered in the current week, each cohort also received additional, different questions. One cohort received spaced practice questions related to the concepts they learned several weeks earlier, while the other cohort received massed practice questions related to the concepts they learned in the current week. Each cohort received some spaced practices and some massed practices throughout the semester so that the sets of questions assigned to each cohort were identical in the end.

Experimental setup. Since the students in Cohorts 1 and 2 receive different sets of questions on Assessment 4, we investigate how this difference affects their learning on the concepts they learn next, i.e., their performance on Assessment 5. Treating each cohort as a PLA, our goal is to maximize the students' performance on Assessment 5 by assigning them to the cohort (selecting the PLA) that benefits them the most. Therefore, in our setting the number of PLAs is $A = 2$. We take the students' graded responses to questions in Assessments 1–3 and apply SPARFA to estimate each student's K-dimensional concept knowledge vector $\mathbf{c}_j$, which we use as the context. We set the number of concepts to $K = 3$.[4] Since Cohorts 1 and 2 also receive different questions for Assessment 5 as part of the spacing vs. mass retrieval practice experiment on new concepts covered in Week 5, we take the set of $Q = 5$ questions shared between the two cohorts to evaluate their performance. Since MAB algorithms analyze students sequentially, we randomly permute the order of the students and average our results over 2000 random permutations.

[4]In our experiments, we have found that the performance of SPARFA and A-CLUB is robust to the number of concepts K as long as $K \ll Q$.

	A-CLUB	LINUCB	UCB
Training set	**3.69**	3.68	3.65
Test set	**3.89**	3.77	3.70

Table 1: Performance comparison of A-CLUB against two baseline algorithms on personalized cohort selection on the physics course dataset. A-CLUB outperforms the other algorithms in terms of average student credit on the follow-up assessment (out of a full credit of 5) on both the training and test sets.

Evaluation method. We use the unbiased offline evaluation approach in [24, 25] to evaluate our algorithms. We use only the students that were actually assigned to the same cohort as chosen by our algorithms and ignore the other students. This approach evaluates the decision making algorithms under the scenario where the data is collected in a specific "off-line, off-policy" manner, i.e., the data is collected by selecting PLAs for each student uniformly at random across every PLA, as opposed to a more typical MAB setting where PLAs are chosen for students sequentially given the observed follow-up assessment performance of previous students. Such a scenario fits our experimental setup well and yields an unbiased estimate of the expected reward for each student [25]. We use the students' total credit on Assessment 5, i.e., $\sum_{i=1}^{Q} s_i Y_{i,j}$, as the metric to evaluate the performance of the algorithms.

Results and discussion. Figure 2 shows the students' average credit (out of a full credit of 5) on Assessment 5 vs. the number of students the algorithms use for the algorithms A-CLUB, LINUCB [24], and UCB [3]. The parameters in every algorithm were tuned for best performance. We see that the average student credit increases as the number of students the algorithms observe increases, i.e., the algorithms improve their PLA selection policy as they see more training data. As a concrete example, by comparing the average student credit at the first and last points on the curves, we see that A-CLUB has found a policy that yields students approximately 10% more credit than a policy that selects PLAs randomly.

Following the approach in [24], we also conduct an experiment by separating the dataset into a training set with 80% of the students and a test set with 20% of the students, to validate both the efficiency (performance on the training set) and efficacy (performance on the test set) of A-CLUB. We train the above three algorithms on the training set and apply the learned PLA selection policy to the test set, and report the average student credit obtained on both sets. Table 1 indicates that A-CLUB outperforms the other algorithms on both the training set and the test set. Better performance on the test set means that A-CLUB learns a better policy than the other algorithms, while better performance on the training set means that it learns this policy very quickly as the amount of training data increases.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a contextual (multi-armed) bandits framework for PLA selection that maximizes students' immediate success on a follow-up assessment, given the latent concept knowledge estimated from their binary-valued graded responses to questions in previous assessments. Our contextual logistic upper confidence bound (CLUB) algorithms learn such a policy and achieve better or comparable performance than baseline algorithms.

There are a number of avenues for future work. First, our context vectors are indexed by student features only, while in the general contextual bandits setting the contexts can be indexed by both student features and features of the learning resources. SPARFA-Trace [22], a recently developed framework for time-varying learning and content analytics, features a mechanism to analyze the content, quality, and difficulty of all kinds of learning resources (i.e., textbook sections, lecture videos, practice questions, etc.). We can apply this approach to extract features from the learning resources that we can integrate into the contexts in our algorithms. Second, we can incorporate an additional PLA that corresponds to "no action", due to the cost of taking actions, as considered in [36]. This extension would enable students with high knowledge on the concepts covered to avoid repeated practice and advance more quickly to new concepts. Third, we are interested in integrating our approach into more sophisticated contextual bandit algorithms, e.g., [37] to reap further performance improvements.

6. ACKNOWLEDGEMENTS

Thanks to Phillip Grimaldi, former pinball champion of Indiana, for collecting the physics course dataset and Mihaela van der Schaar for insightful suggestions. Visit our website `www.sparfa.com`, where you can learn more about the SPARFA project and purchase SPARFA t-shirts and other merchandise.

7. REFERENCES

[1] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320, Dec. 2011.

[2] P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *J. Machine Learning Research*, 3:397–422, Mar. 2003.

[3] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, May 2002.

[4] T. Barnes and J. Stamper. Toward automatic hint generation for logic proof tutoring using historical student data. In *Proc. 9th Intl. Conf. on Intelligent Tutoring Systems*, pages 373–382, June 2008.

[5] Y. Bergner, S. Droschler, G. Kortemeyer, S. Rayyan, D. Seaton, and D. Pritchard. Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. In *Proc. 5th Intl. Conf. on Educational Data Mining*, pages 95–102, June 2012.

[6] P. Brusilovsky and C. Peylo. Adaptive and intelligent web-based educational systems. *Intl. J. Artificial Intelligence in Education*, 13(2-4):159–172, Apr. 2003.

[7] A. C. Butler, E. J. Marsh, J. Slavinsky, and R. G. Baraniuk. Integrating cognitive science and technology improves learning in a STEM classroom. *Educational Psychology Review*, 26(2):331–340, June 2014.

[8] H. Cen, K. R. Koedinger, and B. Junker. Learning factors analysis – A general method for cognitive model evaluation and improvement. In *Proc. 8th Intl. Conf. on Intelligent Tutoring Systems*, pages 164–175, June 2006.

[9] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction*, 21(1-2):137–180, Jan. 2011.

[10] B. Clement, D. Roy, P. Oudeyer, and M. Lopes. Multi-armed bandits for intelligent tutoring systems. *J. Educational Data Mining*, 7(2):20–48, 2015.

[11] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-adapted Interaction*, 4(4):253–278, Dec. 1994.

[12] Coursera. https://www.coursera.org/, 2016.

[13] edX. https://www.edx.org/, 2016.

[14] L. Fahrmeir and H. Kaufmann. Consistency and asymptotic normality of the maximum likelihood estimator in generalized linear models. *The Annals of Statistics*, 13(1):342–368, Mar. 1985.

[15] S. Filippi, O. Cappe, A. Garivier, and C. Szepesvári. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*, pages 586–594, Dec. 2010.

[16] Y. Gong, J. E. Beck, and N. T. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *Proc. 10th Intl. Conf. on Intelligent Tutoring Systems*, pages 35–44, June 2010.

[17] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2010.

[18] M. Khajah, R. Wing, R. Lindsey, and M. Mozer. Integrating latent-factor and knowledge-tracing models to predict individual differences in learning. In *Proc. 7th Intl. Conf. on Educational Data Mining*, pages 99–106, July 2014.

[19] K. R. Koedinger, J. R. Anderson, W. H. Hadley, and M. A. Mark. Intelligent tutoring goes to school in the big city. *Intl. J. Artificial Intelligence in Education*, 8(1):30–43, 1997.

[20] K. R. Koedinger, E. Brunskill, R. S. Baker, E. A. McLaughlin, and J. Stamper. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, 34(3):27–41, 2013.

[21] A. S. Lan and R. G. Baraniuk. A contextual bandits framework for personalized learning action selection – Extended version. Technical report, Rice University, 2016.

[22] A. S. Lan, C. Studer, and R. G. Baraniuk. Time-varying learning and content analytics via sparse factor analysis. In *Proc. 20th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pages 452–461, Aug. 2014.

[23] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. *J. Machine Learning Research*, 15:1959–2008, June 2014.

[24] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proc.19th Intl. Conf. on World Wide Web*, pages 661–670, Apr. 2010.

[25] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proc. 4th ACM Intl. Conf. on Web Search and Data Mining*, pages 297–306, Feb. 2011.

[26] R. Lindsey, M. Mozer, W. Huggins, and H. Pashler. Optimizing instructional policies. In *Advances in Neural Information Processing Systems*, pages 2778–2786, Dec. 2013.

[27] Y. Liu, T. Mandel, E. Brunskill, and Z. Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In *Proc. 7th Intl. Conf. on Educational Data Mining*, pages 161–168, July 2014.

[28] F. Lord. *Applications of Item Response Theory to Practical Testing Problems*. Erlbaum Associates, 1980.

[29] T. Mandel, Y. Liu, S. Levine, E. Brunskill, and Z. Popovic. Offline policy evaluation across representations with applications to educational games. In *Proc. Intl. Conf. on Autonomous Agents and Multi-agent Systems*, pages 1077–1084, May 2014.

[30] OpenStaxTutor. https://openstaxtutor.org/, 2016.

[31] W. Powell. *Approximate Dynamic Programming: Solving The Curses of Dimensionality*. John Wiley & Sons, 2007.

[32] A. N. Rafferty, E. Brunskill, T. L. Griffiths, and P. Shafto. Faster teaching by POMDP planning. In *Proc. 15th Intl. Conf. on Artificial Intelligence in Education*, pages 280–287, June 2011.

[33] A. N. Rafferty, M. M. LaMar, and T. L. Griffiths. Inferring learners' knowledge from their actions. *Cognitive Science*, 39(3):584–618, Apr. 2015.

[34] M. D. Reckase. *Multidimensional Item Response Theory*. Springer, 2009.

[35] J. Rollinson and E. Brunskill. From predictive models to instructional policies. In *Proc. 8th Intl. Conf. on Educational Data Mining*, pages 179–186, June 2015.

[36] C. Tekin, J. Braun, and M. van der Schaar. eTutor: Online learning for personalized education. In *Proc. 40th IEEE Intl Conf. on Acoustics, Speech and Signal Processing*, pages 5545–5549, April 2015.

[37] C. Tekin and M. van der Schaar. RELEAF: An algorithm for learning and exploiting relevance. *IEEE J. Selected Topics in Signal Processing*, 9(4):716–727, June 2015.

[38] K. VanLehn, C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, and M. Wintersgill. The Andes physics tutoring system: Lessons learned. *Intl. J. Artificial Intelligence in Education*, 15(3):147–204, Aug. 2005.

[39] D. Vats, C. Studer, A. S. Lan, L. Carin, and R. G. Baraniuk. Test size reduction via sparse factor analysis. *Preprint*, June 2014.

[40] B. P. Woolf. *Building Intelligent Interactive Tutors: Student-centered Strategies for Revolutionizing E-learning*. Morgan Kaufman Publishers, 2008.

How Good Is Popularity? Summary Grading in Crowdsourcing

Haiying Li
Rutgers University
10 Seminary Place
New Brunswick, NJ 08901
1-848-932-0868
haiying.li@gse.rutgers.edu

Zhiqiang Cai
University of Memphis
365 Innovation Dr.
Memphis, TN 38152
1-901-678-2364
zcai@memphis.edu

Arthur C. Graesser
University of Memphis
365 Innovation Dr.
Memphis, TN 38152
1-901-678-2364
grasser@memphis.edu

ABSTRACT
In this paper, we applied the crowdsourcing approach to develop an automated popularity summary scoring, called wild summaries. In contrast, the golden standard summaries generated by one or more experts are called expert summaries. The innovation of our study is to compute LSA (Latent Semantic Analysis) similarities between target summary and wild summaries rather than expert summaries. We called this method CLSAS, i.e., crowdsourcing-based LSA similarity. We evaluated CLSAS by comparing it with other approaches, Coh-Metrix language and discourse features and LIWC psychometric word measures. Results showed that CLSAS alone could explain 19% of human summary score, which was equivalent to the variance explained by dozens of language and discourse features and/or the word features. Results also showed that adding language and/or word features to CLSAS increased small additional correlations. Findings imply that crowdsourcing-based LSA similarity approach is a promising method for automated summary assessment.

Keywords
Summary grading, Crowdsourcing, LSA Similarity, Coh-Metrix, LIWC

1. INTRODUCTION
The use of the summarization strategy enables to improve reading comprehension and production of expository texts for both L1 learners [1] and L2 learners [2]. Summarizing involves reading processes and reproducing processes. Reading process requires the learners to identify the main ideas and distinguish the important points from the unimportant points. Reproducing process requires the learners to restate the important ideas in a coherent, precise and accurate manner in their own words [3]. Learners' summarizing skill depends on the ability to construct a coherent mental model of the text, which is aligned with text discourse [4]. This ability consists of three knowledge components: rhetorical text structures and genres, propositional text content, and a coherent mental model for a variety of genres [4], which are important for reading comprehension [5]. Summarization strategy is an effective instructional strategy [6] to help students improve these abilities [7] and summary writing is therefore considered as a good measure of reading comprehension at a deep level.

Grading summaries are time-consuming and costly for teachers, so it is impossible for teachers to provide a real-time and instant summary score, let alone provide the instant feedback on the quality of summaries. Researchers thereby have developed the automated summary assessments with the techniques of natural language processing and machine learning [4,8]. These assessments are not practical for teachers because they require model building based on human expert summaries as the reference summaries and a large amount of human summary grading. Thus, model rebuilding is time-consuming and costly for teachers. Each time teachers need to repeat such complex steps as expert-written summaries as reference, human-scored summary as the training set, model training, and model evaluation. As summary writing is a weekly assignment for middle school and high school students, summary grading will be a common task for teachers. The present automated summary assessments will not reduce but increase the teachers' workload. These methods are impractical for teachers to use. Teachers need a more efficient and effective summary assessment with least efforts.

In this paper, we applied the crowdsourcing approach to develop an automated "popularity" summary scoring. Crowdsourcing enables a diverse and a large amount of population to generate abundant summaries, which are called "popularized summaries" or "wild summaries." In contrast, the golden standard summaries generated by one or more experts are called "expert summaries." The innovation of our study is to compute LSA (Latent Semantic Analysis) similarities between the target summary and the wild summaries instead of expert summaries. We called it CLSAS, namely, crowdsourcing-based LSA similarity. We proposed CLSAS was a robust measure for summary grading.

This study makes innovative contributions to the automated summary assessment for three reasons. First, it is efficient and effective, because the model was built based on one feature rather than dozens of features. Second, it is unnecessary for human experts to generate the golden summaries on each quality level. The model was built based on the wild summaries generated by all of the summary writers. Third, it is unnecessary for human experts to manually grade summaries for the model training.

The next section briefly reviews research on automated summary assessment, crowdsourcing approach, and three advanced text analysis tools, LSA similarity [9], Coh-Metrix [10] and LIWC (Linguistic Inquiry and Word Count) [11].

1.1 Automated Summary Assessment
Techniques of natural language processing and machine learning have been used to develop the automated summary assessment [4,8]. Diverse features used in the assessment range from semantic features measured by LSA [8] to language features exacted by BLEU (Bilingual Evaluation Understudy) [4], ROUGE (Recall-

Oriented Understudy for Gisting Evaluation) [12], TERp (Translation Error Rate Plus) [4], and N-gram [12]. Some features were used to detect plagiarism in summary (e.g., N-gram [4]), assess coherence of the summary (e.g., LSA [8] and N-gram [12]), evaluate content unit (e.g., unigram overlap [8]), or examine the length of summary [4]. These assessments were proved to robustly predict human summary grading [4,8] but had the following limitations.

First, all of these assessments need reference summaries that are generated by one or more human experts [4,8]. The reference summaries have different qualities, ranging from good to poor on multiple-point scales [4]. The student's summary is graded by comparing with the reference summaries. The similarities could be computed by similarities of LSA [8], a lexical and phrasal overlap (e.g., ROUGE) [8], N-gram overlap (e.g., BLEU) [4,8], summary length [4], or token count [4]. Second, the sufficient amount of human-graded summaries at each quality level is required to build the model for the supervised learning. Third, different language and discourse features and algorithms are tested in order to build a better fit model. As these assessments are not content independent, these three cycles are repeated if summaries' source text changes. These tasks definitely increase extra workload for teachers, so it is hard and impractical to spread these approaches. It is necessary to develop a summary assessment without expert reference summaries, human grading, and model rebuilding for a new source text. This study aims to explore a real-time and efficient summary assessment that requires the least efforts so that teachers can easily use it by themselves.

1.2 Crowdsourcing

Crowdsourcing refers to a process that mobilizes a huge amount of population (called crowd workers) to accomplish the complex, collaborative, and sustainable tasks on demand and at large scale, especially from an online community rather than traditional employees or suppliers [13]. Crowd workers can either be volunteers for collective projects such as Wikipedia or paid via platform such as Amazon's Mechanical Turk, one popular crowdsourcing platform [13]. Crowdsourcing is frequently used to generate ideas and break down creative tasks into smaller pieces [13-17]. The application of crowdsourcing is an emerging approach in research. For example, some researchers asked crowd workers to create or retrieve content for new stories [16,17], to generate a story [14] or summaries of social media events [15]. This collaborative work provides an author diverse ideas or contents quickly [13-17].

1.3 LSA Similarity

LSA [18] is a mathematical and statistical technique that represents knowledge about words, sentences, paragraphs, and documents on the basis of a large corpus of texts. LSA reduces a large corpus of texts to 100–300 dimensions using singular value decomposition technique. The conceptual similarity between two texts is computed as the geometric cosine between the vectors representing two texts. The cosine value varies from -1 to 1 [18,19], with the higher score representing higher similarity.

LSA is used to assess coherence in Coh-Metrix [10] and quality of essays [8, 20-22]. In addition, LSA has been utilized in the intelligent tutoring system (ITS) to assess the constructed response or the open response, such as AutoTutor [19]. These assessment systems for essay, summary, or open response requires expert reference summaries and human-graded summaries generated by human experts. Few studies do not use expert summaries as reference. Summarization in machine translation develops a fully automated approach to evaluate ranking systems that requires no expert summaries [8]. However, it requires a large amount of content annotations and is restricted to the ranking system, which it is not appropriate for teachers to use for summary grading. Cai et al. [9] explored the LSA similarity model without the golden standard reference for the open response assessment. Instead, the reference was all the responses written by students except the target response. We borrowed this approach in this study and use the learners' summaries as the reference summaries.

1.4 Coh-Metrix

Coh-Metrix (cohmetrix.com) is a computer-based tool that automates many language- and text-processing mechanisms over hundreds of measures of cohesion, language, and readability [10]. Coh-Metrix is developed based on a multilevel theoretical framework [23]. This framework specifies six theoretical levels: words, syntax, explicit textbase (e.g., explicit propositions, referential cohesion), situation model (also called mental model), discourse genre and rhetorical structure (the type of discourse and its composition), and the pragmatic communication level. The first five of these six levels have metrics captured in the Coh-Metrix automated text analysis tool [10].

The current version of Coh-Metrix [10] extracts 110 measures, which are categorized into genre (narrative versus informational), LSA space (e.g., text cohesion), word information (e.g., familiarity, concreteness, imageability, meaningfulness, age of acquisition), word frequency, part of speech, density score (e.g., density of pronouns), logic operators (e.g., *if-then*), connectives (e.g., *therefore*), type/token ratio, polysemy and hypernym, syntactic complexity (e.g., noun phrase density), readability (e.g., Flesch-Kincaid grade level), co-reference cohesion (e.g., noun overlap, argument overlap), along with five primary components extracted based on these features (e.g., narrativity, word concreteness, syntactic simplicity, referential cohesion, and deep cohesion).

1.5 LIWC

LIWC (Linguistic and Inquiry Word Count) [11] computes the percentage of words in a text that fit into the linguistic or psychological categories. The 2015 LIWC dictionary contains 6,400 words, word stems, and select emoticons. It generates 93 measures that are categorized into the following categories: word count, summary language variables (e.g., analytical thinking, authentic, emotional tone), linguistic dimensions (e.g., functional words, pronouns, conjunctions), other grammar (e.g., common verbs, interrogatives), psychological processes (e.g., affective, social, cognitive, informal language). The word count function of LIWC attempts to match each word in a given text to a word in the various categories.

The LIWC categories have been confirmed as valid and reliable markers of a variety of psychologically meaningful constructs [11]. The different categories of words would be expected to predict psychological dimensions. For example, negative emotion words would be diagnostic of gloomy texts. The function words (particularly pronouns) are diagnostic of social status, personality, and various psychological states. Differences in function word use can be reflected by gender, age, and social class. LIWC is used to measure the formal versus informal language formality [24,25].

This paper combined the crowdsourcing approach with the LSA similarity to assess summaries. This approach was evaluated by comparing the Coh-Metrix language and discourse features and

the LIWC word features with the human-graded summary scores as the criteria. Specially, seven models were trained and compared their predictability for the human summary scores: (1) CLSAS, (2) Coh-Metrix language feaures (94), (3) LIWC word features (93), (4) Coh-Metrix + LIWC, (5) CLSAS + Coh-Metrix, (6) CLSAS + LIWC, and (7) CLSAS + Coh-Metrix + LIWC. It is necessary to clarify that the human-graded summary scores were only used to evaluate but not build the model. We hypothesize that crowdsourcing-based LSA similarity is an efficient, effective, and reliable measure for summary grading for the following two reasons. First, LSA is a most robust feature for semantic meaning [11] than the language and word features. Second, the wild summaries as reference maximally represent diversity of students' summaries as compared with expert summaries.

2. METHOD

2.1 Participants

Crowd workers (N = 201) volunteered for 3-hour monetary compensation ($30) on Amazon Mechanical Turk (AMT), a trusted and commonly used data collection service [21]. The basic requirement for participation is that they have the goal to improve English summary writing. Participants were required to complete writing 8 summaries, but only 1,481 summaries were collected due to the technical issues. 71% participants were Asian, 16% white or Caucasian, 7% African American, 5% Hispanic, 2% other. Their average age was 33.50 (SD = 8.79), 57% were male, and 81% with bachelor degree or above.

2.2 Materials

Participants read 8 expository texts with different topics and text difficulties in the AutoTutor CSAL. CSAL is an intelligent tutoring system that teaches adult learners the summarization strategies in order to improve their reading comprehension [19]. Participants were required to write a summary with 50-100 words for each text. Four texts are on comparison-contrast text structure and another four on cause-effect text structure (See Table 1). The text difficulty was measured with the Coh-Metrix formality (z-score) at the multiple textural levels and Flesch-Kincaid grade level, sensitive to word length and sentence length [24]. These 8 texts were formal and above grade 8 to early college grades [24]. The balanced Latin-square designs were applied to control for order effects in terms of text difficulty, topics and text structures.

2.3 Summary Grading

The summaries were graded based on four components: topic sentence, content, grammar and mechanics, and signal words. Table 2 lists the detailed descriptions for three scales of each component, from 0 (minimum) to 2 (maximum) points. Thus, the total score ranged from 0 to 8. Four English native researchers graded summaries, 1 male and 3 female. There were three rounds of training for summary grading and after each grading, and then the disagreements were discussed. Before grading, they got familiar rubrics and then they started the three-round grading with one per week. Each round included 32 randomly-selected summaries (4 from each text and 8 texts in total). Inter-rater reliability was assessed by the intraclass correlation coefficient with a two-way random model and absolute agreement type. The average inter-rater reliability reached the threshold: Cronbach's a = .82, intraclass correlation coefficient = .80. As the average of reliabilities for three training sets were high, each grader graded summaries for two texts in the same text structure.

Table 1. Source Texts and the Number of Summaries (N).

Structure	Topics	Formality	FKGL	Words	N
Comparison	Butterfly and Moth	.12	8.6	255	183
	Hurricane	.20	9.4	222	185
	Walking and Running	.18	8.9	399	187
	Kobe and Jordan	.14	9.2	299	187
Causation	Floods	. 47	9.2	230	186
	Job Market	.62	10.9	240	181
	Effects of Exercising	.28	9.1	195	189
	Diabetes	.64	11.7	241	182

Table 2. Rubrics for Scoring Summary

Categories	2 points	1 points	0 point
Topic Sentence	A clear topic sentence that states the main idea.	A topic sentence that touches upon the main idea.	The summary does not state the main idea.
Content	Major details stated economically and arranged in a logical order. No minor or unimportant details or reflections.	Some but not all major details stated and not necessarily in a logical order. Some minor or unimportant details or reflections.	Few major details stated and not necessarily in a logical order. Many minor or unimportant details or reflections.
Mechanics and Grammar	Few or no errors in mechanics, usage, grammar or spelling.	Some errors in mechanics, usage, grammar or spelling that to some extent interfere with meaning.	Serious errors in mechanics, usage, grammar or spelling, which make the summary difficult to understand.
Signal Words	Uses the clear and accurate signal words to connect information.	Uses several clear and accurate signal words to connect information.	Uses several clear signal words to connect information.

2.4 Measures

In this study, we employed three approaches to assess summaries: semantic meaning measured by LSA similarity, Coh-Metrix, and LIWC. The crowdsourcing-based LSA similarity score was the LSA cosine between a target summary and all the wild summaries from the corresponding source text. 94 language and discourse features were utilized to train and build the Coh-Metrix summary assessment model. All of 93 psychometric word features were utilized to train and build the LIWC summary assessment model.

2.5 Procedure

Participants took a demographic survey, a pretest (1 comparison and 1 causation), training (2 comparisons and 2 causations), and a posttest (1 comparison and 1 causation). On tests, participants wrote summaries by themselves. During training, two agents first interactively presented the importance of signal words for two text structures (comparison and causation) and how to use signal words to identify the corresponding text structure. Then participants interacted with the conversational agents to learn a summarizing strategy with adaptive scaffolding. Participants were required to write a summary with 50 to 100 words for each text. If the amount of words was beyond the range, the agents reminded the participants of the required length. If the participants copied the original sentences with 10 consecutive words, the agents reminded them of using their own words. Agents did not provide the adaptive feedback for their summary writing, but commented on three summary examples with good, medium, and bad qualities for each source text. The primary interface during training was shown in Figure 1.

Note. (A) the teacher agent, Cristina (female), (B) the student agent, Jordan (male), (C) the instruction of the presented question, (D) the text presented with the scroll down button, (E) an input text-box for participants to enter and submit their summaries, and (F) the self-paced next button.

Figure 1. Screenshot of Learning Interface.

3. RESULTS

A series of linear regressions with 10-fold cross-validation in WEKA was performed on 7 models, respectively. Fisher z was used to compare the difference between two pairs of correlations (see Table 3). Results revealed that crowdsourcing-based LSA similarity robustly predicted human summary grading (r = .44; R2 = .19), as well as 55 Coh-Metrix measures (r = .43; R2 = .18), 57 LIWC measures (r = .47; R2 = .22), and 108 measures by Coh-Metrix (57) and LIWC (51) jointly (r = .46; R2 = .21). This indicates that the variance explained by one LSA similarity measure is equivalent to the variance explained by more than 55 language features or word features, and more than 100 language and word features jointly.

Adding 94 Coh-Metrix features to CLSAS added an additional variance (r = .51; R2 = .26) in explaining human grading scores. Adding 93 LIWC features also added an additional variance (r = .55; R2 = .30). Adding both Coh-Metrix and LIWC feature added an additional variance (r = .49; R2 = .24), but the increased variance was significantly lower than by adding either Coh-Metrix or LIWC features. Due to the limited pages and the significant predictors in the Coh-Metrix + LIWC model overlapped with those in the Coh-Metrix model or the LIWC model, we only reported the predominant predictors in the Coh-Metrix model and LIWC model as below.

The 55 Coh-Metrix measures consisted of 9 descriptive (e.g., word count, sentence length), 4 referential cohesions (e.g., noun overlap, argument overlap), 5 LSA overlap (e.g., adjacent sentences, LSA given, LSA new), 3 lexical diversity (e.g., type-token ratio), 5 connectives (e.g., logical, additive), 3 situation model (e.g., causal verbs and particles, LSA verb overlap), 5 syntactic complexity (e.g., minimal edit distance, sentence syntax similarity), 4 syntactic pattern density (e.g., noun phrase density, verb phrase density), 16 word information (e.g., noun, adjective, hypernymy for nouns), and 1 readability (e.g., Flesch Kincaid Grade Level).

The 57 LIWC features consisted of 3 summary variables (e.g., analytical thinking, authentic), 3 language metrics (e.g., sentence length, words with more than 6 letters), 11 function words (e.g., personal pronouns), 4 grammar other (e.g., regular verb, quantifiers), 4 affect words (e.g., emotion words, anger), 3 social words (e.g., friend, gender referents), 3 cognitive processes (e.g., tentativeness, certainty), 3 perceptual (e.g., seeing, hearing), 3 biological processes (e.g., body, health), 2 core drives and needs (affiliation and risk focus), 1 relativity, 4 personal concerns (e.g., religion, home), 2 informal speech (swear and filler), and 3 all punctuations (e.g., apostrophes, comma).

Table 3. Fisher's *z*: Comparisons of Correlations

Models	1	2	3	2+3	1+2	1+3
1 (*r*=.44)	---					
2 (*r*=.43)	-0.34	---				
3 (*r*=.47)	1.03	1.36	---			
2+3 (*r*=.46)	0.68	1.02	-0.35	---		
1+2 (*r*=.51)	2.46**	2.80**	1.43	1.78*	---	
1+3 (*r*=.55)	3.97***	4.31***	2.94**	3.29**	1.51	---
1+2+3 (*r*=.49)	1.74*	2.07*	0.71	1.05	-0.73	-2.24*

Note. 1 = LSA similarity; 2 = Coh-Metrix features; 3 = LIWC features. * $p < .05$. ** $p < .01$. *** $p < .001$.

4. DISCUSSION

This paper developed an effective and efficient automated summary assessment, called crowdsourcing-based LSA similarity (CLSAS). Crowdsourcing enables a diverse and a mass of people to produce abundant wild summaries. CLSAS used the wild summaries rather than the human expert summaries as the reference when computing LSA similarities. The CLSAS was validated by comparing with Coh-Metrix language features, LIWC word features, and both language and word measures together with human-scored summaries as the criteria. Results indicated that CLSAS measure predicted human summary grading as well as over 55 language measures, 57 word measures, and 108 language and word measures, respectively. Even though adding language features, word features, or both to CLSAS improved the predictability, the predictability of CLSAS alone is most robust with correlation coefficient above 6.74 in each model. Findings imply that crowdsourcing-based LSA similarity approach is a promising method and will have good popularity in automated summary assessment.

One possible explanation for the significant predictability of CLSAS is that the wild summaries generated by diverse populations display diverse qualities as compared with few expert summaries. These wild summaries maximally represent the target summary. On the hand, the wild summaries represent neutralized or averaged semantic meaning, which is called *centroid*. The centroid might better capture the semantic meaning represented in

the target summary. For example, the CLSAS model showed that LSA similarity had a very high coefficients, $\beta = 8.60$, which was substantially higher than other measures' in other models.

The Coh-Metrix measures are different from the crowdsourcing-based LSA similarity due to its nature on measuring cohesion, language, and readability rather than semantic meaning [10]. One semantic measure of LSA similarity between the target summary and the crowdsourcing-based summaries is equivalent to 55 Coh-Metrix language measures. Among these language measues, LSA overlap among all sentences in paragraph reached 5.43 for mean and 2.07 for standard deviation; LSA given/new -3.60 for mean and -2.39 for standard deviation; and LSA overlap between adjacent sentences, -1.20 for mean. The other measures showed very low coefficients, generally below 1.00. This implies that a range of language measures jointly plays a role in assessing summaries, but LSA measures are attributed more than others.

Besides the predominant role of LSA measures, other important Coh-Metrix measures included lexical diversity ($\beta = 3.92$) measured by type-token ratio. Type-token ration is widely used for both automated essay assessment [19] and automated summary assessment [4]. When the type-token ratio is high, namely, more unique words are used, the lexical diversity is high and the text is likely to be either very low in cohesion or very short. Oppositely, when the type-token ration is low, namely, more words are repeatedly used, the lexical diversity is low, but cohesion is high. Summarizing requires conciseness and briefness, so in one summary, repeatedly using the same word will lower the quality of summary. Another two crucial measures are sentence syntax similarity between adjacent sentences ($\beta = 4.49$) and across paragraphs ($\beta = -4.71$). The high syntax similarity between adjacent sentences suggests the uniformity and consistency of the syntactic construction. This implies that the whole summary is consistent in syntactic construction. However, the low syntax similarity across paragraphs results in greater syntactic variety.

Another two most robust predictors are paragraph count ($\beta = -12.74$) and word length (number of syllables; $\beta = 4.32$). These two measures are frequently used in the automated summary [4] and essay assessment [19]. Our study controlled the number of words of summaries, which explains why word count is not a robust predictor, as compared with the previous studies [9]. As the summary should be brief and concise, more paragraphs demonstrate the poor quality in conciseness. However, the high word length increases difficult to read and represents an academic or formal language style [25] in the summary.

The phenomena that the Coh-Metrix features were unevenly weighted did not occur in the LIWC features. Specifically, among Coh-Metrix measures, the measures such as cohesion, syntactic and lexical complexity are more robust than measures at the word level. LIWC measures are all at the word level, but go beyond the linguistic words. They expand to diverse psychometric words, such as analytical thinking, emotion, and social. All the LIWC measures are evenly weighted to predict human summary scores. This pattern occurs in the Coh-Metrix and LIWC joint model as well. These findings suggest that each type of words plays a small piece of role, as compared to language and semantic measures.

Fisher's *z* comparisons CLSAS with Coh-Metrix measures, LIWC measures, and Coh-Metrix + LIWC measures demonstrated no differences in explained variance in human summary grading between CLSAS and Coh-Metrix, CLSAS and LIWC, and CLSAS and Coh-Metrix + LIWC. The findings supported our hypothesis that CLSAS could predict human summary grading as well as dozens of language measures and/or LIWC measures.

To further evaluate the validity of CLSAS, we added Coh-Metrix, LIWC, and Coh-Metrix + LIWC measures to CLSAS model with different combinations. Results showed adding each of these features increased the predictability. It is easier to explain the incremented model because the language and word features represent different aspects of summary assessment and enable to compensate the semantic feature. No matter what features were added to CLSAS, CLSAS is consistently the most significant feature in the models. Specifically, the correlation coefficient of LSA was 7.49, 6.74, and 6.80 when adding the Coh-Metrix language features, the LIWC word features, and both, respectively. Therefore, LSA similarity was a robust feature for summary assessment, no matter when it is used alone or jointly with other features.

5. CONCLUSION

These findings suggest that crowdsourcing-based LSA similarity (CLSAS) is a robust predictor of human summary grading and it is a reliable measure for the automated summary assessment, as compared with a range of language and word measures. As CLSAS has a powerful predictability for human summary score, the wild summaries are assumed as a promising and encouraging approach to replace the expert summaries for its time-saving and efficient. Opposed to the tedious and time-consuming manual summary grading, the wild summaries have no doubt for its popularity and practicability for teachers. This efficient and effective summary grading could dramatically encourage and motivate the teachers to instruct the summarization strategy. Consequently, this will enhance the students' summarization skills, especially summary writing. For example, when teachers need to grade the students' summaries, they could use all of the summaries that the students wrote as the reference. These summaries wildly generated by the students represent diverse qualities. For a particular target summary, the teacher only clicks the target summary and its CLSAS will be automatically computed with all of the summaries. Each time teachers need summary grading, they could repeat this cycle, no any human grading is needed. Based on the LSA similarity score, the summary score could be generated.

This crowdsourcing approach could be popularized and applied to the ITS learning and assessment environment as well. The current ITS assessment assesses the open response with a list of stored expectations and misconceptions [19]. Unfortunately, students' answers could not be assessed accurately due to the unmatched "golden" reference. To address this issue, the crowdsourcing generated responses could be adopted as the reference to replace the limited number of responses that the human expert generates. However, the reliability and validity of the wild open responses need to be evaluated in the future research.

The future study should concentrate on scaling crowdsourcing-based LSA similarity score into 3- or 5-point scales that teachers usually use for a better interpretation. The present study only showed its predominant role in summary assessment without specifying the extent to which LSA similarity score represents the different levels of summaries. The present study compared the CLSAS approach with dozens of measures, which may have an overfitting problem. The future study could select the most popular features that are used in the automated summary assessment and compared them with the CLSAS approach.

To sum up, this study proposed an innovative approach, crowdsourcing-based summary assessment, to the summary assessment from two perspectives. First, the summary reference could be a range of summaries that are wildly generated by a lot of population who are not necessary to be experts. Second, LSA similarity between the target summary and the wildly-generated summaries is a powerful predictor for human summary grading. This innovation will advance the development of automated assessment, especially automated assessment in the ITS.

6. ACKNOWLEDGMENTS

The research reported in this paper was supported by the National Science Foundation (0325428, 633918, 0834847, 0918409, 1108845) and the Institute of Education Sciences (R305A080594, R305G020018, R305C120001, R305A130030).

7. REFERENCES

[1] McNamara, D. S., Crossley, S. A., & McCarthy, P. M. (2010). The linguistic features of quality writing. *Written Communication, 27*, 57–86. DOI= 10.1177/0741088309351547.

[2] Crossley, S. A., Salsbury, T., McNamara, D. S., and Jarvis, S. 2010. Predicting lexical proficiency in language learner texts using computational indices. *Language Testing, 18*, 561-580. DOI= 10.1177/0265532210378031.

[3] Kintsch, E. (1990). Macroprocesses and microprocesses in the development of summarization skill. *Cognition and Instruction, 7*(3), 161-195. DOI= 10.1207/s1532690xci0703_1.

[4] Madnani, N., Burstein, J., Sabatini, J. and O'Reilly, T., 2013. Automated scoring of a summary writing task designed to measure reading comprehension. NAACL/HLT 2013, 163.

[5] Kintsch, W., 1998. Comprehension: A paradigm for cognition. Cambridge university press.

[6] Friend, R., 2001. Effects of strategy instruction on summary writing of college students. *Contemporary Educational Psychology, 26*(1), 3-24. DOI= 10.1006/ceps.1999.1022.

[7] G. Yu. 2003. Reading for summarization as reading comprehension test method: Promises and problems. *Language Testing Update, 32*:44–47.

[8] Passonneau, R. J., Chen, E., Guo, W. and Perin, D. 2013. Automated pyramid scoring of summaries using distributional semantics. In *ACL* (Sofia, Bulgaria, August 4-9, 2013), 143-147.

[9] Cai, Z., Graesser, A.C., Forsyth, C., Burkett, C., Millis, K., Wallace, P., Halpern, D. and Butler, H., 2011. Trialog in ARIES: User input assessment in an intelligent tutoring system. In *Proceedings of the 3rd IEEE International Conference on Intelligent Computing and Intelligent Sy*stems (San Francisco, C.A., August 4-6, 2011), 429-433

[10] McNamara, D. S., Graesser, A. C., McCarthy, P. M., & Cai, Z. (2014). Automated evaluation of text and discourse with Coh-Metrix. New York: Cambridge University Press.

[11] Pennebaker, J.W., Boyd, R.L., Jordan, K. and Blackburn, K., 2015. *The development and psychometric properties of LIWC2015*. UT Faculty/Researcher Works.

[12] Lin, C.Y. and Hovy, E. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1* (Edmonton, Canada, May 27-June 1, 2003). Association for Computational Linguistics, 71-78

[13] Kittur, A., Nickerson, J.V., Bernstein, M., Gerber, E., Shaw, A., Zimmerman, J., Lease, M. and Horton, J. 2013. . The future of crowd work. In *Proceedings of the Conference on Computer Supported Cooperative work* (Antonio, TX, February23-27, *2013*), ACM, 1301-1318

[14] Kim, J., Cheng, J. and Bernstein, M.S. 2014. Ensemble: exploring complementary strengths of leaders and crowds in creative collaboration. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative work & Social Computing* (Vancouver, BC, March 14-18, 2014). ACM, 745-755

[15] Kim, J. and Monroy-Hernandez, A., 2015. *Storia: Summarizing social media content based on narrative theory using crowdsourcing*. arXiv preprint arXiv:1509.03026.

[16] Matias, J.N. and Monroy-Hernandez, A., 2014, . NewsPad: Designing for collaborative storytelling in neighborhoods. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems* (Totonto, Canada, April 26-May 01, 2014). ACM, 1987-1992.

[17] Agapie, E. and Monroy-Hernandez, A., 2015. Eventful: Crowdsourcing Local News Reporting. arXiv preprint arXiv:1507.01300.

[18] Landauer, T. K., McNamara, D., Dennis, S., and Kintsch, W. (Eds.). (2007). Handbook of latent semantic analysis. Mahwah, NJ: Erlbaum.

[19] Li, H., Shubeck, K., and Graesser, A. C. (2016). Using technology in language assessment. In D. Tsagari and J. Banerjee (Eds.), Contemporary second language assessment. London, UK: Bloomsbury Academic.

[20] Landauer, T.K., Laham, D. and Foltz, P.W. 2003. Automatic essay assessment. *Assessment in Education: Principles, Policy & Practice, 10*(3), 295-308.

[21] Burstein, J. 2003. The E-rater scoring engine: Automated essay scoring with natural language processing. In M. D. Shermis and J. C. Burstein (Eds.), *Automated essay scoring: A cross-disciplinary perspective*. Mahwah, NJ: Erlbaum, 113-122.

[22] Nenkova, A. and Passonneau, R., 2004. Evaluating content selection in summarization: The pyramid method.

[23] Graesser, A. C., & McNamara, D. S. (2011). Computational analyses of multilevel discourse comprehension. *Topics in Cognitive Science, 3*, 371-398. DOI: 10.1111/j.1756-8765.2010.01081.x.

[24] Graesser, A.C., McNamara, D.S., Cai, Z., Conley, M., Li, H. and Pennebaker, J., 2014. Coh-Metrix measures text characteristics at multiple levels of language and discourse. *The Elementary School Journal, 115(2)*, 210-229. DOI= 10.1086/678293.

[25] Li, H., Graesser, A.C., Conley, M., Cai, Z., Pavlik Jr, P.I. and Pennebaker, J.W., 2015. A New Measure of Text Formality: An Analysis of Discourse of Mao Zedong. *Discourse Processes*, 1-28. DOI= 10.1080/0163853X.2015.10101.

Beyond Log Files: Using Multi-Modal Data Streams Towards Data-Driven KC Model Improvement

Ran Liu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
ranliu@cmu.edu

Jodi Davenport
WestEd
300 Lakeside Drive, 25th Floor
Oakland, CA 94612
jdavenp@wested.org

John Stamper
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jstamper@cs.cmu.edu

ABSTRACT
The increasing use of educational technologies in classrooms is producing vast amounts of process data that capture rich information about learning as it unfolds. The field of educational data mining has made great progress in using log data to build models that improve instruction and advance the science of learning. Thus far, however, the predictive and explanatory power of such models has often been limited to the actions that educational technologies can log. A major challenge in incorporating more contextually rich data streams into models of learning is collecting and integrating data from different sources and at different grain sizes. We present our methodological advances in automating the integration of log data with additional multi-modal (e.g., audio, screen video, webcam video) data streams. We also demonstrate several examples of how integrating multiple streams of data into the knowledge component (KC) model refinement process improves the predictive fit of student models and yields important pedagogical implications. This work represents an important advancement in facilitating the integration of rich qualitative details of students' learning contexts into the quantitative approaches characteristic of EDM research.

Keywords
Multi-Modal Data Analytics, KC Model Improvement, Log Data, Structured Event Analysis of Multiple Streams (SEAMS)

1. INTRODUCTION
As student learning becomes increasingly conducted on computers and other digital devices, vast amounts of learning-related data are produced. Ideally, such data will provide a rich picture of student knowledge and behaviors (e.g., [8]). But predicting performance and generating pedagogical insight is limited, in the majority of cases, to the actions that digital systems can log. Computerized tutors are often used in a classroom context, and log data cannot capture all learning phenomena. A student working at a computer might be working independently with few outside influences. Alternatively, she might be in a lively classroom, with other students around her, talking and even offering suggestions. Data that capture the context surrounding educational technology use may add to and complement log data. In some cases, it may lead to critical insights.

Educational data mining analyses often omit additional contextual data for a number of reasons. Data on classroom context are difficult to collect. Data from different sources are often collected at different grain sizes, which are difficult to integrate. Here, we present work that extends educational data mining techniques to incorporate multiple modalities of data (computer log files, audio, screen videos, and webcam videos). We present methods we developed that help streamline both the collection of additional streams of data and the linkage across multiple streams. In two experiments, we then demonstrate the value of incorporating multi-modal, contextually rich data streams into established educational data mining techniques. In the first experiment, students use a chemistry virtual lab tutor and, in the second, students use an intelligent tutoring system to collaborate on fraction arithmetic.

Specifically, we extend methods of data-driven knowledge component (KC) model refinement [17] by incorporating, into the process, multiple streams of data spanning different modalities. We show that KC model improvements uniquely derived from these additional data beyond log files led to improved predictive models of student learning and behavior. These improved models of learning, in turn, can generate actionable knowledge for systems, students, teachers, and researchers.

2. BACKGROUND
2.1 Related Work
Recent work reflects a growing interest in multi-modal data analytics, particularly surrounding project-based, constructionist, and/or informal learning contexts [4, 18]. These efforts have focused on capturing divergent student strategies [4] and interactions that happen outside of a traditional computer tutor environment (e.g., with peers and with the physical environment [16]). Their primary goal is to make technologies supporting open-ended learning environments more scalable and to develop assessments appropriate for this type of learning.

Areas of research within the EDM community have also focused on collecting sources of data computer logs cannot capture to serve as "ground truth" labels in training log-data based detectors. These efforts have largely focused on modeling and detecting students' motivational and affective states [2, 8, 15]. For example, models can detect patterns of log data activity that precede affective states like confusion, frustration, and boredom. Physiological data may also be collected and used to develop models that can detect affective states from machine-readable signals, such as facial features, body movements, and electrodermal activity [14].

Outside of these pockets of the community, though, the majority of EDM research has focused exclusively on using log data to model learning. Building statistical models to predict step-level performance and data-driven KC model (or Q-matrix [3]) discovery are examples of major branches of EDM research that are typically limited to computer-logged data. In the present work, we demonstrate the value of expanding EDM research to include additional data streams that convey important contextual information about students' learning. We also present methodological advancements that improve the ease with which

additional data streams can be collected and incorporated into educational data mining methods more broadly.

2.2 Data-Driven KC Model Improvement

Knowledge component models are an important basis for the instructional design of automated tutors and are important for accurate assessment of learning. Knowledge components (KCs) refer to units of knowledge representation (e.g., facts, concepts, or skills) that students need in order to solve problems. A KC Model maps a set of KCs mapped to a set of items or problem steps. Student models that are based on more accurate KC models produce better predictions of what a student knows based on their performance and, thus, result in better assessment and improved learning and instruction [11]. Cognitive Task Analysis is the traditional method for creating cognitive models of learning, but it requires subjective decisions and large amounts of human time and effort. Data-driven techniques of KC model discovery and refinement, when applied to large sets of educational data, can provide both more objectivity and reduce human effort.

A method developed by [17] leverages tools available in the PSLC DataShop [10] to identify potential improvements to a KC model in a data-driven manner. This method iterates through the following steps: (1) inspect learning curve visualizations and best-fitting statistical parameter estimates for the best existing KC model, (2) identify problematic KCs, (3) hypothesize changes to the KC model based on examining constituent problem content and applying domain expertise, and (4) re-fit the statistical model with the revised KC model and assess improvements in predictive accuracy. The premise for this method is that a hallmark of learning on a well-defined KC is a smooth learning curve that shows monotonic improvement in performance over time. KCs that lack these learning curve characteristics, but not because students are at ceiling performance, are likely to involve certain problem steps that require unlabeled difficulty factors or knowledge demands.

After a problematic KC is identified, its constituent problem steps must be examined in order to identify potential hidden difficulty factors. Thus far, this part of the process is limited to what computer log data. For example, a researcher might examine the error rates of the different constituent problem steps for the KC in question and the problem step names to gain clues about hidden difficulties. In the best-case scenario, the researcher might have access to the actual problem content for the dataset (as in [17]) and can apply domain knowledge to identify potential KC modifications. This step of content examination can be greatly enriched by additional streams of contextually rich data from the relevant moments of learning. To this end, we present a method of integrating streams of contextual audio and video data into the KC model refinement process. We show that such integration leads to insights that would not be derived by solely analyzing log data or curriculum content in isolation. We present several examples of how these insights lead to quantitative KC model improvements that improve the overall fit of student models to the data.

3. METHODS

We developed a method of semi-automatically extracting epochs, across multi-modal data streams, associated with the moments during which students engage with a particular KC of interest. This allows the content reviewer, after identifying a candidate KC, to not only view the curriculum content associated with a given KC but also to experience students engaging with that curriculum content through multiple modalities.

There are many ways to collect additional streams of contextually rich data (e.g., using video cameras, external microphones, eye-trackers, and sensors). We focused on a method that minimizes both deployment effort and interference with students' usage of educational technology to increase the likelihood that researchers would consider collecting, analyzing, and sharing such data. In the following experiments, we used Camtasia to simultaneously capture audio recordings, screen videos, and webcam videos of the students. Camtasia can be run in the background to collect all of these streams of data while a student engages with educational software. We installed Camtasia to all classroom laptops in advance of the two studies. On each day of the studies, we opened Camtasia and prepared recording settings before each class period so that all students needed to do was click a red "Record" button prior to logging into the tutors. At the end, students were led through a simple sequence of steps to ensure that their recordings were saved and named properly for easy post-hoc identification.

All recordings (audio, screen video and webcam video) for a single session are initially saved in a Camtasia-specific file format. We used the batch processing function to import and convert the original files to MP4 files that contained all data streams merged. We used timestamp information within the log files to map segments of log data to the appropriate corresponding multi-modal video stream. This step required human input, as Camtasia does not automatically log the system time (at millisecond level) that marks the start of the video recording. For each video file, someone must identify the offset between the beginning of the video and the time of some event in the log file. This offset can then be used to automatically align all remaining events between the log file and the corresponding video files.

We developed a tool called Structured Event Analysis of Multiple Streams (SEAMS) that builds upon the moviepy Python package in conjunction with the FFMPEG multimedia framework to automatically extract video epochs associated with specific events in the log data. The tool allows the user to indicate any event type that can be identified by labels within the log data and generates a folder of video clips that contain all epochs of the merged data streams pertaining to the particular event of interest (in this case, a specific KC at the specific opportunity count). With the relevant epochs grouped together in a manner that allows for quick and effortless analysis by a human examiner, it becomes much easier to quickly view multi-modal data streams to identify hidden knowledge demands towards KC refinement.

We applied our methods to examine the contributions of additional multi-modal data streams on KC model refinement across two classroom experiments. One experiment engaged students in a Chemistry Virtual Lab tutor for which we collected both screen videos and webcam data of learners' facial expressions in addition to traditional log data. The other experiment engaged students in a Collaborative (partner-based) Fraction tutor, and we collected screen videos and audio recordings of students' collaborative dialogue. Due to processor limitations of the school laptops that were available for the Collaborative Fraction tutor experiment, we were not able to collect webcam data. Using the data from both of these studies, we illustrate the application of our methods to leverage the additional multi-modal data streams to improve upon existing KC models. These KC model improvements, in turn, yielded insights about how to improve instruction within the respective tutors.

4. EXPERIMENTS

4.1 Chemistry

ChemVLab+ (chemvlab.org) provides a set of high school chemistry activities designed to build conceptual understanding and inquiry KCs [6]. In each activity, students work through a series of tasks to solve an authentic problem and receive immediate, individualized tutoring. As students work, teachers are able to track student progress throughout the activity and attend to students that may be lagging behind. Upon completion of the activities, students receive a report of their proficiency on targeted KCs, and teachers can view summary reports that show areas of mastery or difficulty for their students. In the current study, students completed four modules: PowerAde: Using Sports Drinks to Explore Concentration and Dilution, The Factory: Using a City Water System to Explore Dilution, Gravimetric Analysis, and Bioremediation of Oil Spills.

4.1.1 Participants

Participants were 59 students at a high school in the greater Pittsburgh area enrolled in honors chemistry classes. They participated in four Stoichiometry modules of the ChemVLab+ educational tutor. They completed these modules across four 50-minute class periods spread over the course of 3 weeks. We collected, using Camtasia, audio recordings and screen video captures for 58 students and webcam recordings of facial expressions for a subset of 25 students who were comfortable with their face being recorded during tutor use.

4.1.2 Results

The newly developed methods facilitated the identification of the way in which a problematic KC needed to be split as well as technical issues that impacted student learning. First, following methods described in [17], we identified a knowledge component called *Concentration* that seemed to have uncharacteristically high error rates on later practice opportunities (Figure 1). This KC represents understanding that the measure of concentration is the amount of substance (e.g., a sports drink powder) in a volume of substrate (e.g., water). It also represents being able to read, report, and compare concentrations of solutions.

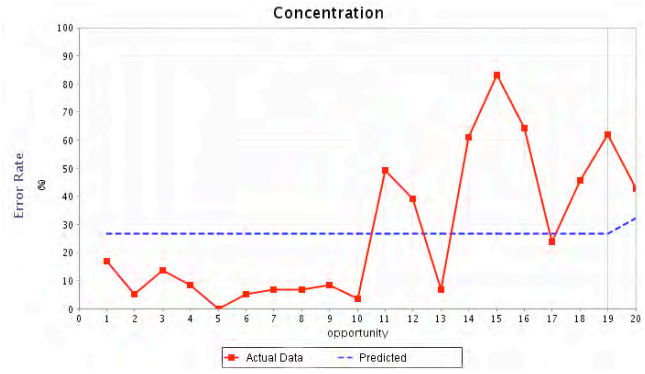


Figure 1. Aggregate learning curve for the *Concentration* KC as originally defined by the ChemVLab+ tutor.

We then used the methods described in Section 3 to automatically extract the screen and webcam videos of all epochs of students engaging with the Concentration KC on their 11th, 12th, 14th, 15th, 16th, and 19th practice opportunities. These were the opportunities on which the KC learning curve had unusually high error rates.

Qualitative analyses of these video stream epochs revealed that students were particularly confused by problems that involved dilution in conjunction with concentration, particularly when a dilution ratio or "factor" is involved. Students demonstrated this confusion as they responded to prompts such as 'Create a 1:2 dilution of the reported sample' or 'Add water to the sample until the concentration is diluted by a factor of 2'. The correct solution requires students to know that the amount of substance (e.g., the powder) takes up negligible volume, so to dilute the powder by 2x, the total amount of water needs to be doubled. Students demonstrated shallow knowledge by responding to prompts like these by adding two parts water to one part solution rather than adding one part water to one part solution, which halves the concentration. In another example, prompt 'Dilute this sample by a ratio of 6:1' student tended to add six parts water to one part of solution (making the resulting amount of powder to volume 1:7), part rather than adding five parts of water to one part of solution (making the resulting amount of powder to volume 1:6).

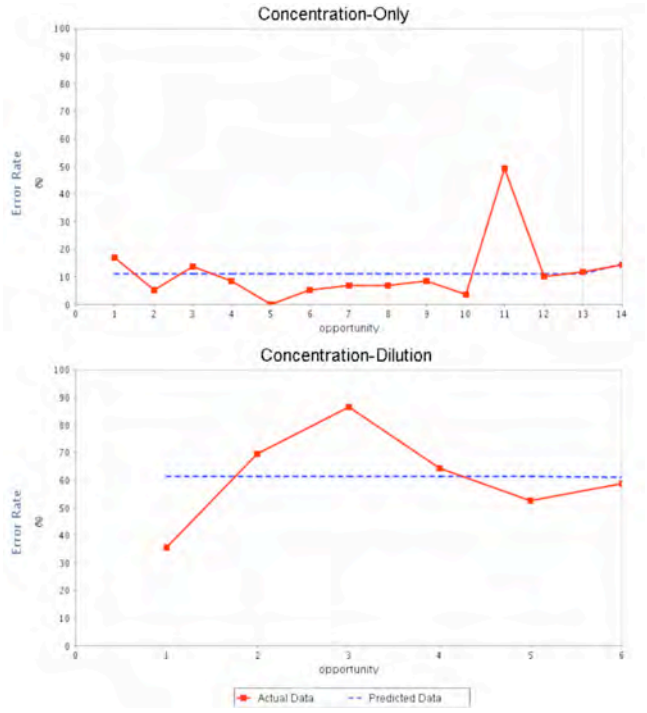


Figure 2. Aggregate learning curves for the two new KCs, *Concentration-Only* and *Concentration-Dilution*, resulting from the KC model refinement process.

Based on this insight, we split the Concentration KC into cases where the problem step required a conceptual understanding of dilution ratios/factors (Concentration-Dilution) and cases where it did not (Concentration-Only). The learning curves for the resulting two KCs are shown in Figure 2. The curves are much smoother than the original learning curve, with the exception of a particular opportunity count with unusually high error rate in the resulting 'Concentration-Only' KC at practice opportunity 11.

To further examine this unusual blip, we re-applied our method to automatically extract screen and webcam videos of all epochs of the 11th opportunity to practice the Concentration-Only KC. We noticed that the majority of problem steps experienced by students on this opportunity count were from a particular screen in the tutor in which the problem text was cut off in the interface. This resulted in students being confused about what they should be doing on this problem. Guessing the answer incorrectly was a common first attempt, as was clicking a hint button. Since the

problem text was fine when viewed on research computers, it did not appear to be a problem with the educational software itself. We hypothesize that the problem may have been due to a unique interaction between the software and the resolution of the computers that students were working on. This is a reality of educational technology deployment in classrooms, and it would have been impossible to know from strictly the log data file or even problem content records that this was the source of students' struggle. If we had only accessed the recorded (idealized) version of the problem content, we may have incorrectly attributed the high error rate on this problem step to intrinsic content present within the problem. After separating these problem steps out from the 'Concentration-Only' KC, the resulting learning curve was much smoother, with an overall low error rate (Figure 3).

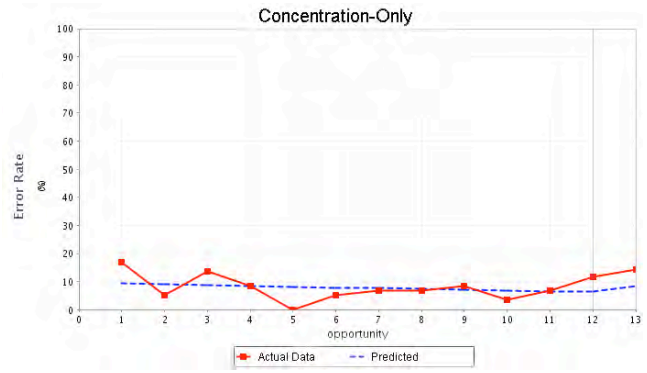


Figure 3. Resulting *Concentration-Only* learning curve, after separating out the problem step in which students experienced a technical difficulty during deployment.

Student model predictive fit metrics are shown in Table 1 for the different KC models when used in conjunction with the Additive Factors Model [5] and reveal an improvement in predictive fit across all metrics (AIC, BIC, and 10-fold cross validation) after splitting the original Concentration KC based on our qualitative analysis of student behavior during epochs of that KC (Row 2). Further improvements in predictive fit across all metrics were observed after we separated out the problem step that contained missing problem text during implementation (Row 3).

Table 1. Student model fit metrics comparing different models resulting from the KC model refinement process.

	AIC	BIC	Cross Validation RMSE
Original KC model	6694.58	7196.59	0.3859
'Concentration'-Split KC Model	6388.35	6904.12	0.3838
'Concentration'-Split KC Model with text-error problem step separated	6318.95	6848.47	0.3819

Both of these KC model refinements, each of which resulted in a substantive and consistent improvement in predictive accuracy when used by the Additive Factors Model, were uniquely dependent on qualitative analyses of the video data we had collected using Camtasia. Although it may have been possible to recognize that the concept of dilution ratios was an additional difficulty factor by purely accessing problem content, there were many other differences between the high error-rate problem steps and the low error-rate problem steps that constituted the original Concentration KC. For example, many of the higher error rate problem steps were part of a different activity (Activity 2, The Factory) than the lower error rate problem steps were (Activity 1, Powerade). Only by observing the students specifically exhibiting actions suggestive of possessing a shallow understanding of dilution ratios (via Camtasia screen videos) and affective states resembling frustration (via webcam videos) were we able to quickly identify the true hidden difficulty factor. Another benefit of this insight, perhaps even more significant than generating a better fitting KC model, is that there are clear implications for instructional redesign. That is, future iterations of the ChemVLab+ tutor might include instruction that more directly targets the misconceptions that students seem to have about the relationship between dilution ratios and existing solutions.

Discovering the high-error-rate problem step in which text was cut off would not have been possible without viewing the real context in which students experienced the problem. Since it was not a general problem with the ChemVLab+ tutor but, rather, an idiosyncrasy in that problem's display on the technology used in the classroom, the Camtasia screen videos were critical in correctly attributing the source of these errors.

4.2 Collaborative Fraction Tutor

The collaborative fraction tutor is online software developed by researchers at Carnegie Mellon University that helps students become better at understanding and working fractions. The tutor was created using Cognitive Tutor Authoring Tools, which allow for rapid development and easy deployment of intelligent tutors [1]. This particular fraction tutor supports collaboration between partners in order to learn fraction-solving KCs such as addition, subtraction, comparing fractions to determine which is larger or smaller, finding the least common denominator, and finding equivalent fractions. In the tutor, each student in a pair can control only part of the screen, so both partners must to work together in order to finish the problem. One student cannot do the whole thing him or herself. Students work at the same time and can talk about what they are doing, ask for help from their partner, and generally collaborate to get the correct answer.

4.2.1 Participants

Participants were 26 fifth grade students at a middle school in the greater Pittsburgh area enrolled in an advanced math class. Students participated across five 45-minute class periods on consecutive days within a week. On the first and last days, students took a computerized pre- and post-test, respectively. They engaged in the Collaborative Fraction Tutor during the three consecutive days between the pre- and post-test days. Students spent half of each class period working individually and half collaborating with a partner. Students were paired with the same person for all partner activities throughout the experiment. We also collected audio and screen video captures for all students working both individually and in pairs on the three tutor use days.

4.2.2 Results

The newly developed methods facilitated the identification of KCs that needed to be split. First, as in [17], we identified a knowledge component called *LCD_procedural* that was noisy, in particular due to an uncharacteristically high error rate on the 5th practice opportunity (Figure 4). We then used the methods described in Section 3 to automatically extract the combined audio and screen videos of all epochs of students engaging in their 5th opportunity of the LCD_procedural KC. Based on qualitative analyses of the

video and audio streams, it was clear that the most common mistake that students were making on those practice opportunities was multiplying the two denominators but failing to reduce the product to find the least common multiple. This was particularly apparent in students' collaborative dialogue following their incorrect first attempts. Students often verbalized the realization that there must be a smaller common multiple. This verbalization did not occur on problems in which the product of denominators happened to be the correct solution. This suggests that there was a separate learning curve for the additional difficulty factor of cases where finding the least common denominator required reducing the product of the two original fractions' denominators to find a smaller common multiple. Based on this, we split the LCD_procedural KC into cases where the LCD required reducing from the product of denominators (LCD_procedural_REDUCE) and cases where it simply was the product of denominators (LCD_procedural_PRODUCT). The resulting learning curves (Figure 5) are much smoother than the original learning curve.

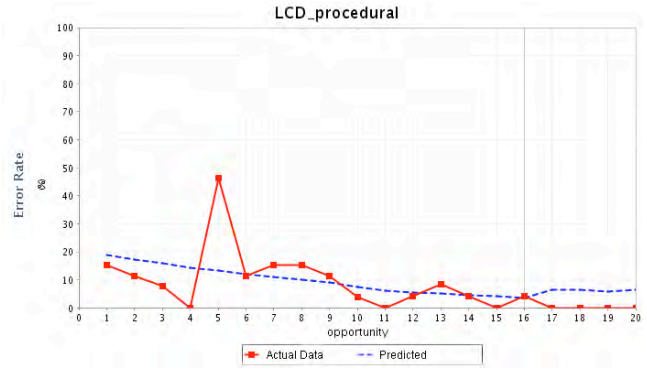


Figure 4. Aggregate learning curve for the *LCD_procedural* KC as originally defined by the Collaborative Fraction tutor.

The student model predictive fit metrics (Table 2) for the different KC models, when used in conjunction with the Additive Factors Model, reveal a substantial improvement in predictive fit across all metrics (AIC, BIC, and 10-fold cross validation) after splitting the original LCD_procedural KC based on our qualitative analysis of student behavior during epochs of that KC.

Through the audio-video segments, we observed students make denominator-product-based errors on their incorrect first attempts and realize they needed to find a smaller common multiple on certain problem steps. This greatly streamlined our identification of the hidden difficulty factor. As a result, we were able to quickly identify the appropriate KC split that led to much smoother learning curves and a better fitting student model.

This discovery also has important instructional implications: for example, the tutor might incorporate a bug message specific to students' inputting the product of the two denominators when the answer is a smaller multiple (i.e., "Can you find a smaller number that divides both denominators?"). A student model based on the revised KC model (with 'LCD_procedural' split into two separate KCs) would also result in students receiving more practice on problems in which the correct answer is a smaller multiple than the product of the two denominators. These instructional changes, resulting from the audio dialogue and video driven insights, will give students better support to overcome this difficulty.

Table 2. Student model fit metrics compared between the original KC model and the improved KC model resulting from multi-modal data stream driven refinement process

	AIC	BIC	Cross Validation RMSE
Original KC model	3497.6	4156.3	0.2738
'LCD_procedural' split KC model	3462.2	4134.5	0.2734

5. DISCUSSION & FUTURE WORK

The vast majority of EDM research, especially research focused on predicting student performance and generating pedagogical insights, is limited to models based on computer-logged data. A recognized issue within the EDM community is that log data cannot capture all learning phenomena; it can miss important details of both learning processes and the learning context. Recent advances in DataShop [10] allow researchers to connect problem names in log data to screenshots of problem content and encourage inclusion of contextual details in custom fields of log data. Clearly, however, there are still instances where a better understanding of the implementation environment and students' experience working through certain problem steps is needed, as demonstrated here.

The main contributions of this work are (1) developing methodological advancements (e.g., the SEAMS tool) that facilitate the ease with which EDM researchers can incorporate context-rich data streams into quantitative modeling techniques, and (2) demonstrating the utility of doing so. Using a top-down,

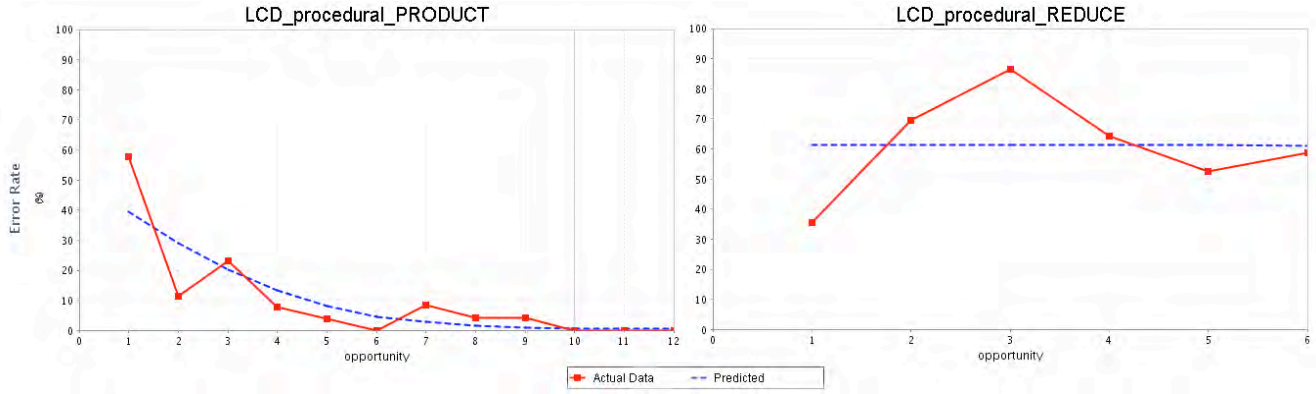


Figure 5. Aggregate learning curves for the two new KCs, *LCD_procedural_PRODUCT* and *LCD_procedural_REDUCE*, resulting from our KC model refinement process5.

KC visualization driven method, we show that valuable qualitative insights can be obtained from targeted segments of audio and video data even without fully "coding" all of the multiple streams. We also show that these qualitative insights lead to quantitative model fit improvements and actionable pedagogical implications.

There are many promising areas for future work based on the methods we have developed here. The present work has focused on refining an existing KC model. Educational data does not always come with an existing expert-labeled KC model, and there have been recent efforts to automatically generate, or discover, KC models [9, 12, 13]. One concern about fully machine-discovered models is their interpretability. The ability to view contextually-rich audio and video segments corresponding to machine-discovered KCs will facilitate the interpretation of these KCs and, in turn, help researchers refine their methods to yield more interpretable or cognitively plausible KC models.

Another interesting issue that contextually-rich streams of data are uniquely suited to address is the attribution of pauses of activity in the log data. A pause in the data because a student is off-task has very different implications than a pause because the student is actively help-seeking outside of the educational technology interface. Being able to use detailed information about students' learning context can help produce correct interpretations of log data activity and, in turn, more robust student models.

Finally, one of the interesting data streams we collected in the Chemistry dataset was student-facing webcam video. Aside from noticing the moments during which students seemed frustrated in the Chemistry tutor due to confused about dilution ratios, we have not yet fully explored the extent to which the webcam data could be used to improve KC models and student models. There is rich potential for our methods to facilitate connections between the cognitive (e.g., knowledge component modeling) and the affective [2, 8] branches of EDM research.

6. ACKNOWLEDGMENTS

We thank Jacklyn Powers and Jenny Olsen for help in collecting the Chemistry and Math tutor data used in our experiments here. This research was supported by the National Science Foundation (Grants #DRL-1418072, PI Davenport and #DRL-1418181, PI Stamper) and the Institute of Education Sciences (Training Grant R305B110003 to Liu). Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF or IES.

7. REFERENCES

[1] Aleven, V., Sewall, J., McLaren, B.M., and Koedinger, K.R. (2006). Rapid authoring of intelligent tutors for real-world and experimental use. In *Proceedings of the 6th ICALT*. IEEE, Los Alamitos, CA, pp. 847-851.

[2] Baker, R.S., Corbett, A.T., Roll, I. and Koedinger, K.R. (2008). Developing a generalizable detector of when students game the system. *UMUAI*, 18(3), pp. 287-314.

[3] Barnes, T. (2005). The q-matrix method: Mining student response data for knowledge. In *Proceedings of the AAAI-EDM Workshop,* pp. 978–980.

[4] Blikstein, P. (2013). Multimodal learning analytics. In *Proceedings of the 3rd International Conf. on LAK*. ACM, New York, NY, pp. 102-106.

[5] Cen, H., Koedinger, K.R., and Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on ITS,* pp. 164-175. Berlin: Springer-Verlag.

[6] Davenport, J., Rafferty, A., Timms, M., Yaron, D., and Karabinos, M. (2012). ChemVLab+: Evaluating a virtual lab tutor for high school chemistry. In *International Conf. of the Learning Sciences*.

[7] D'Mello, S.K., and Calvo, R. (2011). Significant Accomplishments, New Challenges, and New Perspectives. In R. A. Calvo and S. D'Mello (Eds), *New Perspectives on Affect and Learning Technologies*. New York: Springer, pp. 255-272.

[8] Graesser, A.C., Conley, M., and Olney, A. (2012). Intelligent tutoring systems. In K.R. Harris, S. Graham, and T. Urdan (Eds.), *APA Educational Psychology Handbook: Vol. 3. Applications to Learning and Teaching*. Washington, DC: American Psychological Association, pp. 451-473.

[9] Gonzalez-Brenes, J.P., and Mostow, J. (2012). Dynamic Cognitive Tracing: Towards Unified Discovery of Student and Cognitive Models. In *Proceedings of the 5th International Conf. on EDM*.

[10] Koedinger, K.R., Baker, R.S., Cunningham, K., Skogsholm, A., Leber, B., and Stamper J. (2010). A Data Repository for the EDM community: The PSLC DataShop. In Romero C, Ventura S, Pechenizkiy M, Baker RSJd (Eds.), *Handbook of Educational Data Mining*. Boca Raton, FL: CRC Press.

[11] Koedinger, K.R., Stamper, J.C., McLaughlin, E.A., and Nixon, T. (2013). Using data-driven discovery of better cognitive models to improve student learning. In *Proceedings of the 16th International Conf. on AIED.*

[12] Lan, A.S., Studer, C., Waters, A.E., and Baraniuk, R.G. (2014). Sparse Factor Analysis for Learning and Content Analytics. *Journal of Machine Learning Research,* 15, pp. 1959-2008.

[13] Lindsey RV, Khajah M, Mozer MC. (2014). Automatic discovery of cognitive skills to improve the prediction of student learning. In *Advances in Neural Information Processing Systems, 27,* pp. 1386-1394.

[14] Picard, R., Vyzas, E., and Healey, J. (2001). Toward machine emotional intelligence: Analysis of affective physiological state. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 23(10), 1175-1191.

[15] Porayska-Pomsta, K, Mavrikis, M, D'Mello, S.K., Conati, C., and Baker, R. (2013). Knowledge elicitation methods for affect modelling in education. *IJAIED*, 22, 107–140.

[16] Schneider, B., and Blikstein, P. (2014). Unraveling Students' Interaction Around a Tangible Interface Using Gesture Recognition. In *Proceedings of the 7th Annual International Conf. on EDM.*

[17] Stamper, J. and Koedinger, K.R. (2011). Human-machine student model discovery and improvement using DataShop. In *Proceedings of the 15th International Conf. on AIED*.

[18] Worsley, M. (2012). Multimodal learning analytics: enabling the future of learning through multimodal data analysis and interfaces. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, pp. 353-356.

Seeking Programming-related Information from Large Scaled Discussion Forums, Help or Harm?

Yihan Lu
School of Computing, Informatics & Decision Systems Engineering, Arizona State University,
699 S. Mill Ave., Tempe AZ, USA
lyihan@asu.edu

I-Han Hsiao
School of Computing, Informatics & Decision Systems Engineering, Arizona State University,
699 S. Mill Ave., Tempe AZ, USA
Sharon.Hsiao@asu.edu

ABSTRACT
Online programming discussion forums have grown increasingly and have formed sizable repositories of problem solving-solutions. In this paper, we investigate programming learners' information seeking behaviors from online discussion forums. We design engines to collect students' information seeking processes, including query formulation, refinement, results examination, and reading processes. We model these behaviors and conduct sequence pattern mining. The results show that programming learners indeed seek for programming related information from discussion forums by actively searching on the site and reading posts progressively according to course schedule topics. Advanced students consistently perform query refinements, examine search results and commit to read, however, novices do not. In addition, advanced students commit to read posts, but novices only skim.

Keywords
Programming; Information Seeking; Hidden Markov Model; Discussion Forums; Sequential pattern mining;

1. INTRODUCTION
In teaching and learning programming, students are typically asked to refer to API (Application Programming Interface) or programming textbooks for relevant information (i.e. code syntax or code examples). In recent years, open & free online communities (such as homework-help sites, discussion forums for MOOCs courses etc.) have grown increasingly and have formed sizable repositories of problem solving-solutions. They are filled with thousands of programming problem-solving tips, such as "*how-to*" questions [1], people-valued examples, and the examples' explanations [2] etc. On the other hand, from a constructive point of view, the action of articulating a problem and initiating search or referencing can also be a valuable learning activity as well as browsing the solution. In software engineering field, such programming information seeking has already been recognized as a core sub-task in software maintenance [3, 4]. Programmers are even being referred as task-oriented information seekers, which they focus on finding the answers they need to complete a task using a variety of information sources [5]. There are tools that have been built to make completing programming tasks easier, such as Mica [6]. However, none of these tools focuses on amplifying learning opportunities if any, rather, centers on task-oriented problem solving facilitation.

In addition, according to Information Foraging theory [7], finding information is human nature. To successfully form information seeking criteria for a given programming problem requires complex cognitive activities (i.e. defining and verbalizing the programming problem; refining query criteria and selecting results; strategies application etc.) To better support information seeking and learning, we focus on learners' behaviors in seeking programming-related information. Specifically, we investigate in an online large-scale discussion forum, StackOverflow, which is one of the biggest online programming Q&A sites communities and currently hosts a massive amount of heterogeneous definitions, solutions and examples of programming languages. Are those assorted content in the forum helpful or harmful for programming learners?

Studies have shown that while there is a positive connection between the usage of StackOverflow and GitHub (open source code management service), StackOverflow's users consider the site to be more attractive and beneficial for learning programming [8]. In recent learning science literature, learning-from-observing paradigm appears to be a promising strategy, which passive participants (such as lurkers who consume content without contributions) can still learn by reading the postings-and-replies exchanges from others due to the constructive responses in the content [9]. Knowledgeable students can benefit from text with cohesive gaps by making active retrieval and inferences [10]. They can also benefit from building memory and fluency through the active retrieval opportunities and to refine the conditions of application through feedback on incorrect solution attempts in problem solving [11]. On the other hand, novices may benefit from seeing examples of solution steps and from seeing the entire solution structure to make sense of the role of each step in order to construct integrated knowledge components for generating plans and sub goals [12]. In this work, our goal is to investigate what are programming learners' tactics in searching for relevant information from online discussion forums and how do they look for relevant learning materials from massive forum posts.

In this paper, we design engines to capture programming learners' activities on StackOverflow site, such as problem verbalization in queries, query revision and other information seeking processes. We collect a semester long of *informal* programming learning activities from programming discussion forum. We model their information seeking activities by using Hidden Markov Model and data mine the post of their readings.

2. LITERATURE REVIEW
2.1 Modeling Information Seeking In Learning
Traditionally, information seeking is associated with behavioral science theories, which focus on seekers' information needs, searching strategies, and how they use the information. For example, self-awareness of one's information needs, self-regulated learning strategies, information searching experience and ability, etc.[13-15]. Puustinen and Rouet [13] further

classified help-seeking behavior into different types on a help-seeking continuum, a function of the helpers' capacity to adapt answers to their needs. In more recent information seeking literature, we see studies show that users commonly exhibit exploratory behavior in a great extent when performing searches [14]. Marchionini [15] identifies a range of search activities that differentiate exploratory search from look up search (i.e. fact-finding retrieval). Such behavior is especially pertinent to learning and investigating activities, which is the targeted area of interest in our research.

2.2 Modeling Learning From Discussion Forums

Over the decades, data mining on discussion forums has been carried out through various formats, network analyses, topical analyses, interactive explorers, knowledge extraction, etc. [16-18]. Due to calculation complexities (since linguistic features rely on computer processing power), most of these in-depth analyses were performed offline [19, 20]. As a result, the lesson learned could only be applied in the next iteration of system development. Recently, however, we begin to see some studies that focus on dynamic support for users [21]. With the rapid growth of free, open, and large user-based online discussion forums, it is essential, therefore, for education researchers to pay more attention to emerging technologies that facilitate learning in cyberspace. For instance, Wise, Speer, Marbouti, and Hsiao [22] studied an invisible behavior (listening behavior) in online discussions, where the participants are students in a classroom instructed to discuss tasks on the platform; van de Sande & Leinhard [23] investigated online tutoring forums for homework help, making observations on the participation patterns and the pedagogical quality of the content; Hanrahan, Convertino & Nelson [24] and Posnett, Warburg, Devanbu, & Filkov [25] studied expertise modeling in a similar sort of discussion environment.

3. METHODOLOGY

3.1 Research Platform & Data Collection

In this project, we deployed a Chrome browser plugin to track users' query, searching, and reading behaviors on StackOverflow (SO). User can search query on StackOverflow and identify their intention with this tool. The browser plugin has two main features. (1) It provides a direct search channel for users to issue queries on StackOverflow; (2) It displays users' search histories. We collect not only users' search queries, but also their search intentions, including "Knowledge seeking", "Method learning", "Problem solving", and "Other" (indicated by the user). Most importantly, we log all the users' behaviors, comprising of scrolls, clicks, selections, and corresponding actions' time. The behavior tracking function resides on StackOverflow site once initial log in via the SO search tool. In another word, all students' behaviors on StackOverflow site will be logged after at least one time log in via SO Search Tool. However, since they issue the queries directly from StackOverflow site, their intention will be marked as "not specified".

3.2 Study Setup

In order to understand the students' information seeking behaviors on discussion forums, we conducted a user study in a programming class in Arizona State University. Students were encouraged to install the browser plugin search tool. They were told that their search activities would be collected via the tool. All students' programming information seeking behavior was logged during the entire semester.

Additionally, we also conducted a controlled session of lab class during the semester. In the lab class, students were instructed to solve a complex task (implement a 3-way merge sort algorithm) by using the information-seeking tool within 75 minutes. All the students' searching and reading behaviors on StackOverflow were recorded.

Students were given a pretest to examine their pre knowledge about programming. In this study, the students are split into two groups (*Novice* & *Advanced*) based on their pretest median score, which is ranged from 0 to maximum score 20.

3.3 Data Descriptive

Among 86 students in the Object-Oriented Programming class, 71 students voluntarily installed our search plugin, whose operations on SO were automatically recorded, 55 of them also used the plugin to search queries. There were 44 of them took the pretest. According to their pretest score distribution, 24 of them were identified as novices, and 20 were classified as advanced students.

3.3.1 Query data log

For these 55 students provided query information, the average query number is 9.55 (max 56, min 1, median 8), and the average number of operations is 7179 (min 1, median 2917, max 140300). In terms of the query content, the average number of words in each query is 3.76, and the number of distinct words is 573. The frequency distribution for each word approximately follows Zipf's law, which states that the relation between the word frequency and its rank is exponential in general. Considering the pre knowledge of students, queries are separate by whether the provider is novice or advanced student. The novices provided more query in average (13.2±11.7) than advanced students (8.9±9.0), but novices' length of each query (3.47±2.01) is shorter than advanced ones (4.62±2.61), which indicated a lower quality according to Belkin's research [28].

3.3.2 Operation data log

There are 466,659 operations logged including *scroll up*, *scroll down, click* and *select* for both searching and reading phases. We found that for both groups of students, novices and advanced students, generated the majority of the operations in reading and in scrolling down. There were 19.3% operations are scrolling up in the searching phase in general, which was not a trivia finding, It showed that users were going back and forward to review the posts content before they decide to click in to proceed further reading in detail, However, ideally a successful search process is that after entering the query, the best item would be shown in the first place of the search result, so that the user would not even need to scroll before clicking to view a result. However in reality, users need to scroll down when they do not feel satisfied with the results provided in the first view, and this unsatisfying ratio is reflected by the scrolling back and forward operation percentage.

On the other hand, the time cost before each operation shows that when browsing search results, users appear to spend more time (37.8%) before clicking or selecting, while they are faster when reading a specific question-answer thread. This fact indicates that users would read more carefully, or be more serious when choosing a thread to read among the search results.

Considering pre knowledge difference, the ratio of scroll back for novices were lower in searching phase compared to the advanced students, but their scroll back ratio is higher in reading phase. This indicates that the novices were more likely to make a choice without browsing more search results, and they had to read the content for more times compare to advanced students.

3.4 Programming Information Seeking Actions

In order to analyze students programming information seeking behavior on discussion forums, we categorize their actions into 6 categories based on Marchionini's [18] information seeking processes: formulate queries, query refinement, results examination, and reading. According to the amount of operations made on each single page, we further split search and reading (by median) in large-search (LS), small-search (SS), large-read (LR), small-read (SR). Table 1 describes detail of user search actions.

Based on the operation data collection and the above action definitions, 2681 actions were identified in total, and the distribution of action distribution is shown in Figure 2.

Table 1. Programming information seeking actions

Actions	Description
Query (Q)	a student issues an query to look for information from programming discussion forum
Refine query (q)	a student modifies the original Q and issues a similar query (word adjacent distance less than 0.3)
Large search (LS)	A student browses the search result page and did operations more than the median of all search pages (31 operations)
Small search (SS)	A student browses the search result page and did operations less than the median
Large read (LR)	A student reads a Q&A thread page, and did operations more than the median of all reading pages (64 operations)
Small read (SR)	A student reads a Q&A thread page, and did operations less than the median

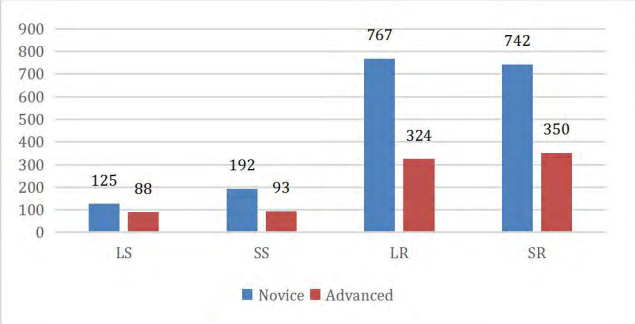


Figure 2. Number of actions identified for novices and advanced students

3.5 Modeling Programming Information Seeking From Discussion Forums Using HMM

The Hidden Markov Model (HMM) is a popular method for modeling sequential data. Previous studies have already shown its ability in modeling user information search process [26], survey design [27] and student learning process [28]. In this study, we employ the HMM to model users' hidden tactics in searching for programming related information on discussion forums, and refer the actions on the site (e.g. query refinement, results examination, content reading, information extraction) as the generated hidden tactics. The hidden tactics can be explained as the strategy used as informal learning activities by looking for programming related information.

We have a sequence of information seeking behaviors from T1 to TM, and each state is one of those predefined information seeking actions: TS = {Q, q, LS, SS, LR and SR}. HMM assumes that we also have a sequence of hidden states, from H1 to HM, and each answer type is generated by a corresponding hidden state, but different answer types can be generated by the same hidden state with different probabilities. A HMM model has several parameters: the number of hidden states HS, the start probability of each states π, the transition probabilities among any two hidden states A_{ij}, and the emission probability from each state to each action b_{ij}. By only defining the HS and π, a Baum-Welch algorithm [29] can be used to learn the emission and transition probabilities.

4. EVALUATION RESULTS

4.1 Mapping HMM Patterns to Information Seeking Processes

In this section HMM is used to detect the students' information seeking behavior pattern. In order to identify the complete sequence of information seeking operations, we only included those operations following a query recorded. The web paged that the students searched from other search engines, where queries were not included, are excluded.

The first step of using HMM is to determine the number of hidden states. A larger number of states will help to describe the model more precisely, while the risk of over-fitting is also increased. In model selection, the information criterion such as the Akaike Information Criterion (AIC) or its variants Bayesian information criterion (BIC) [29] can be used to determining the optimal number of states. Based on models best performance by AIC, we choose HS=3 and HS=5 for *Advanced* and *Novice* groups accordingly (Figure 3).

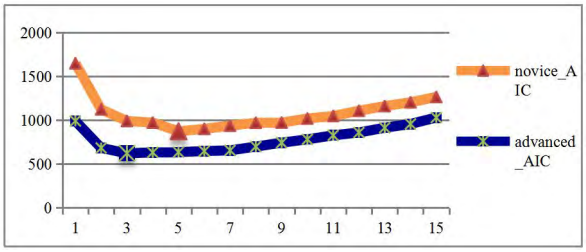


Figure 3. Choosing number of hidden state using AIC.

The emission probability of each hidden state to information seeking operations is shown in Table 2, in which the probabilities under 0.05 were removed for better presentation of the results. The hidden states can be treated as the underlying "tactics" or "principles" when students look for programming information from the discussion forum. For example, *Advanced* group HS2 demonstrates the stronger students' reading behaviors, which they appear to do more careful readings and fast browsing; while in *Novice* group HS3, students tend to perform more superficial reading than careful reading. While *advanced* group shows more coherent searching, browsing and reading behaviors (each behavior is observed by single state), novices show duo searching and browsing behaviors. *Novice* HS4 and HS1 states seem to have similar searching and browsing behaviors as *advanced* group. However, *Novice* HS5 exhibits more distinct searches by issuing queries and lower probability in refining queries. In addition, Novice HS2 shows high probabilities in small search, which can be interpreted as careless results examination.

Table 2. The hidden states of programming information seeking operations (b_{ij})

hidden states		Q	q	LS	SS	LR	SR
Advanced	HS1	0	0	0.39	0.61	0	0
	HS2	0	0	0	0	0.79	0.22
	HS3	0.76	0.24	0	0	0	0
Novice	HS1	0	0	0.36	0.64	0	0
	HS2	0	0	0.05	0.95	0	0
	HS3	0	0	0	0	0.35	0.65
	HS4	0.73	0.27	0	0	0	0
	HS5	0.85	0.15	0	0	0	0

Figure 4 is plotted according to the transition probability, and the prior probability is shown in Table 3. The probabilities under 0.05 are removed. HS3 has the highest prior probability (start probability) in *advanced* group, which means that advanced students always begin with issuing query and modifying the query. So do the majority of the weaker students. In addition, HS5 state is also another beginning state with high probability for novices. It shows that there is also a great probability that novices start issuing queries with minimal query refinement. However, what are the impacts of the amount of query refinement? We have to look at what is happening next. According to Figure 4, the *Advanced* & *Novice* state transition diagrams, there are several findings listed below:

Table 3. The prior probability of each hidden state (π)

	HS1	HS2	HS3	HS4	HS5
Advanced	0	0	**1**	-	-
Novice	0	0	0	**0.536**	**0.464**

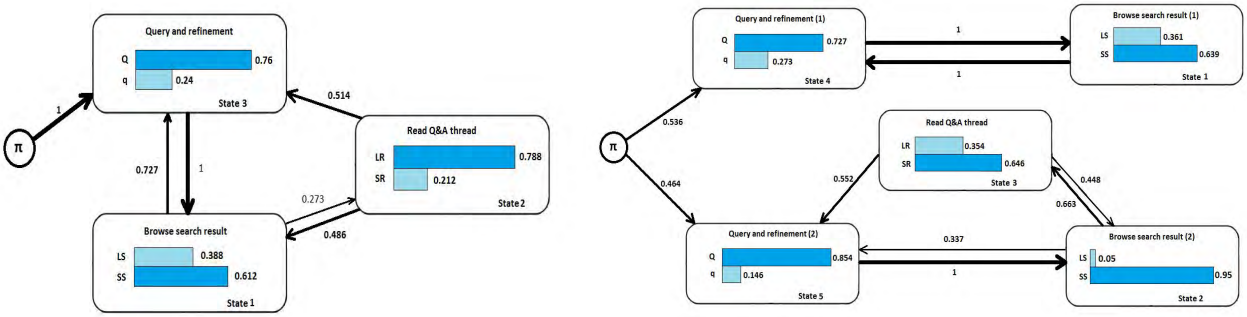


Figure 4. *Advanced* (left) and *Novice* (right) students' information seeking transition probability diagrams

4.1.1 Advanced students refine query; novices don't

Advanced students consistently performed query refinements (3:1 ratio) before they examine the results (HS3 → HS1). Novices behaved differently. Part of them followed the similar pattern as *Advanced* students did, tuning the queries before examine the results (HS4 → HS1). However, when these novices refined queries, there were no consecutive actions followed in the next step (Figure 4 – right top), which indicated that they did not go to any reading page. On the other hand, when novices did minimum query refinements (HS5 → HS2), they did manage to proceed to next step, which was the reading phase (HS5 → HS2 → HS3). This fact suggested that novices may lack of query-results examination ability and lead to no reading (HS4 → HS1). In addition, as the HS2 of *Novice* group shows, 95% of the likelihood that the operations were small searches, which means that novices tended not to scrutinize the search results, they only examined the results minimally, even move on to read forum posts (HS5 → HS2 → HS3). They could read whatever the discussion forum has recommended (i.e. top returned items).

In fact, Table 4 shows the total amount of time that each student spent on searching or reading pages. It is surprising to see that novices spent more than 130 minutes on just reading, while advanced students spent about 40 minutes. Similarly, novices spent more time on searching compare to advanced students. The reason of the time difference is not only they browsed more pages, but also their time spent on each page is longer. These findings indicate that the novices' searching and browsing behaviors only consist of minimum query refinement so that they had to spend more time to read and understand search results, which can be due to insufficiency of vocabulary in searching and lack of judgment in finding reading resources. We further looked into students' reading behavior and reading content in the following section. Despite the reading quality, novices' behaviors can also suggest the *hidden danger* of online large-scale discussion forums, where the existing filtering mechanisms (such as badges, acceptance, and votes) may not be enough, especially for novice learners.

Table 4. Total time spent on searching and reading average per student

total time (seconds) / student	Novice (N=24)	Advanced (N=20)
Search	340.5	146.4
Read	7870.3	2366.6

4.1.2 Advanced students read and novices skim

When students eventually landed on forum post pages and read, we found that *advanced* students committed to careful reading, while novices did more skimming (*Advanced* HS2: 0.79 LR; *Novice* HS3: 0.65 SR). In fact, we found that novices cost more time in small reading than advanced students, while in large reading advanced students spent slightly more time, but there was no significant difference between groups. These results reveal that novices performed less reading in search results filtering, but once they did, they would spend time to read. Thus, it led us to examine their learning effect. Do novices and advanced students have similar effects after reading?

4.2 Reading and Learning Effects

4.2.1 Students read posts according to course schedule topics

In order to understand what content were students' reading, we crawled all the posts that students read from StackOverflow, and performed text mining with MALLET[1] LDA toolkit with default α=30/N, β=0.01, *itr*=1000. We found students were reading the contents from discussion forums according to the course weekly topics, from week 1 *Java Basis* to week 9 *LinkedList*. We then used all the topic words generated from the LDA model to compute Shannon entropy score in estimating the topic focus (Figure 5). There are several interesting findings: *Advanced* students were generally more focused across all topics (smaller topic entropy), except week 4 and week 9. The effect was much more apparent in complex topics: *Recursive* (Table 5 shows the extracted topic words, which we found advanced students read posts regarding to a specific recursive implementation Fibonacci sequence, which novices did not). In week 4 and 9, advanced students were found to be less focused in terms of reading more diverse topics was due to those two weeks were exam periods. Therefore, it is understandable that students might read a wider range of topics that were covered over exam periods.

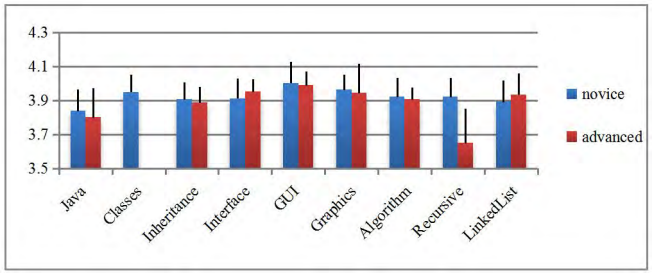


Figure 5. Weekly readings' keywords by novices and advanced students

Table 5. Recursive topic words by novices and advanced students

Novice: {type, code, recursive, dynamic, void, write, result, example, loop, print, add, wikipedia, error, int, version, method, operator, pseudo, easy, program, static, mathematics, call, line, learn, number, work, value, function, undefined}
Advanced: {function, method, value, static, return, int, change, version, recursive, result, error, mathematics, program, line, number, fibonacci, sequence, fib, wikipedia, operator, pseudo, easy, type, print, example, code, learn, void, traverse, loop}

4.2.2 Learning Effects

Based on the percentage of large read rate in reading pages, we found that the more students spending time in reading on StackOverflow, the higher final score they obtained (r=0.418, p<0.01). Additionally, we found that the slope of novices and advanced students had little difference, while the intercept of novices is higher. This fact indicates that novice and advanced students gained the same benefits from increasing large read rate, however, in order to achieve the same score, novices has to read more carefully. Figure 6 shows the connection between large read rate and final exam score.

[1] http://mallet.cs.umass.edu

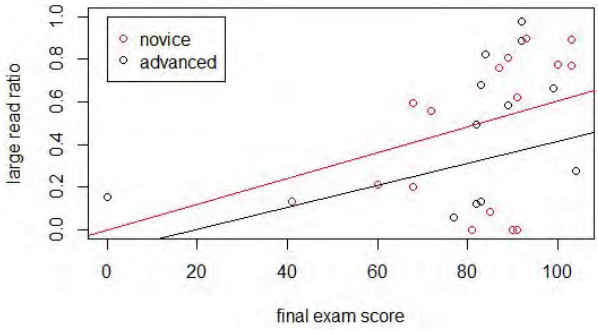


Figure 6. Final score vs. Large read rate

5. CONCLUSIONS

5.1 Summary

In this study, we designed a programming information seeking framework with a browser plugin to collect students' programming information seeking behavior data from discussion forum StackOverflow. Students' query intention, time spent and all actions were logged. We modeled programming learners' query formulation, refinement, results examination, and reading processes with Hidden Markov Model. We conducted sequence pattern mining. The results showed that programming learners indeed seek for programming related information from discussion forums by actively searching on the site and reading posts progressively according to course schedule topics.

The result of this study showed that programming novices usual spend more time in browsing search result and reading, while the sequential due to their lack of pre knowledge. As long as they can read as well as advanced students, they can learn as much as advanced students according to the learning evaluation result.

All the study results shed lights on programming learners seek for learning resources from large-scale online discussion forums. We anticipate this work serves as guidelines for educational technologists to design better effective tools to facilitate learning via programming information seeking process.

5.2 Limitations and Future Work

There are a few limitations in current study. First of all, after students log in from the browser at least once, all their activities on StackOverflow will be recorded. However, when students search from search engines (i.e. Google) and land on StackOverflow site, their initial queries will not be captured. A more completed data collection should include all queries that the students search in information seeking.

Moreover, we mainly take into account of students' query and mouse actions without considering other keystrokes' actions. Another common information seeking behavior is to use Ctrl+F on the keyboard to search keyword with in a web page, which was not captured in the study. This operation can be a convenient and fast method to locate useful information when browsing web pages, including discussion forums.

In the future, we will consider a more completed data collection and more exhaustive evaluation. Most importantly, we aim to design an adaptive programming information seeking tool to help novices effectively navigate search results.

6. REFERENCES

[1] Vasilescu, B., Serebrenik, A., Devanbu, P., & Filkov, V. (2014, February). How social Q&A sites are changing

knowledge sharing in open source software communities. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing* (pp. 342-354). ACM.

[2] Treude, C., O. Barzilay, and M. Storey. How do programmers ask and answer questions on the web?: NIER track. in *Software Engineering (ICSE), 2011 33rd International Conference on*. 2011.

[3] Seaman, C.B. The information gathering strategies of software maintainers. in *Software Maintenance, 2002. Proceedings. International Conference on*. 2002. IEEE.

[4] Sharif, K.Y. and J. Buckley. Developing schema for open source programmers' information-seeking. in *Information Technology, 2008. ITSim 2008. International Symposium on*. 2008. IEEE.

[5] Sim, S.E., *Supporting multiple program comprehension strategies during software maintenance*. 1998, University of Toronto.

[6] Stylos, J. and B.A. Myers. Mica: A Web-Search Tool for Finding API Components and Examples. in *Visual Languages and Human-Centric Computing, 2006. VL/HCC 2006. IEEE Symposium on*. 2006.

[7] Kuhlthau, C.C., Inside the search process: Information seeking from the user's perspective. *Journal of the American Society for Information Science*, 1991. 42(5): p. 361-371.

[8] Eickhoff, C., Teevan, J., White, R., & Dumais, S. (2014, February). Lessons from the journey: a query log analysis of within-session learning. In *Proceedings of the 7th ACM international conference on Web search and data mining* (pp. 223-232). ACM

[9] Chi, M.T.H. and R. Wylie, The ICAP Framework: Linking Cognitive Engagement to Active Learning Outcomes. *Educational Psychologist,* 2014. 49(4): p. 219-243.

[10] McNamara, D. S. (2001). Reading both high-coherence and low-coherence texts: Effects of text sequence and prior knowledge. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 55(1), 51.

[11] Matsuda, N., Cohen, W. W., Sewall, J., Lacerda, G., & Koedinger, K. R. (2008, June). Why tutored problem solving may be better than example study: Theoretical implications from a simulated-student study. In *Intelligent Tutoring Systems* (pp. 111-121). Springer Berlin Heidelberg.

[12] Catrambone, R. (1996). Generalizing solution procedures learned from examples. *Journal of Experimental Psychology: Learning, Memory, and Cognition,* 22(4), 1020.

[13] Puustinen, M. and J.-F. Rouet, Learning with new technologies: Help seeking and information searching revisited. *Computers & Education*, 2009. 53(4): p. 1014-1019.

[14] Zimmerman, B.J. and M.M. Pons, Development of a Structured Interview for Assessing Student Use of Self-Regulated Learning Strategies. *American Educational Research Journal*, 1986. 23(4): p. 614-628.

[15] Marchionini, G., Exploratory search: from finding to understanding. *Communications of the ACM*, 2006. 49(4): p. 41-46.

[16] Dave, K., M. Wattenberg, and M. Muller, Flash forums and forumReader: navigating a new kind of large-scale online discussion, in *Proceedings of the 2004 ACM conference on Computer supported cooperative work*. 2004, ACM: Chicago, Illinois, USA. p. 232-241.

[17] Indratmo, J. Vassileva, and C. Gutwin, Exploring blog archives with interactive visualization, in *Proceedings of the working conference on Advanced visual interfaces*. 2008, ACM: Napoli, Italy. p. 39-46.

[18] Guerra, J., Sahebi, S., Lin, Y. R., & Brusilovsky, P. (2014). The problem solving genome: Analyzing sequential patterns of student work with parameterized exercises. in *The 7th International Conference on Educational Data Mining. 2014*: London, UK.

[19] Wen, M., D. Yang, and C. Rose. Sentiment Analysis in MOOC Discussion Forums: What does it tell us? in *the 7th International Conference on Educational Data Mining*. 2014. London, UK.

[20] Wang, X., Yang, D., Wen, M., Koedinger, K., & Rosé, C. P. Investigating how student's cognitive behavior in MOOC discussion forums affect learning gains, in *The 8th International Conference on Educational Data Mining*. 2015: Madrid, Spain.

[21] Enamul Hoque, G.C., Shafiq Joty. Interactive Exploration of Asynchronous Conversations: Applying a User-Centered Approach to Design a Visual Text Analytic System. in *Workshop on Interactive Language Learning, Visualization, and Interfaces*. 2014. Baltimore, Maryland.

[22] Wise, A. F., Speer, J., Marbouti, F., & Hsiao, Y. T. (2013). Broadening the notion of participation in online discussions: examining patterns in learners' online listening behaviors. *Instructional Science*, 41(2), 323-343.

[23] Sande, C.v.d., Free, open, online, mathematics help forums: the good, the bad, and the ugly, in *Proceedings of the 9th International Conference of the Learning Sciences - Volume 1*. 2010, International Society of the Learning Sciences: Chicago, Illinois. p. 643-650.

[24] Hanrahan, B.V., G. Convertino, and L. Nelson, Modeling problem difficulty and expertise in stackoverflow, in *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion*. 2012, ACM: Seattle, Washington, USA. p. 91-94.

[25] Posnett, D., Warburg, E., Devanbu, P., & Filkov, V. (2012, December). Mining stack exchange: Expertise is evident from initial contributions. In *Social Informatics (SocialInformatics), 2012 International Conference on* (pp. 199-204). IEEE.

[26] Han, S., Z. Yue, and D. He. Automatic detection of search tactic in individual information seeking: A hidden Markov model approach. in *iConference 2013*. 2013. arXiv preprint arXiv:1304.1924.

[27] Hsiao, I. H., Han, S., Malhotra, M., Chae, H. S., & Natriello, G. (2014, June). Survey sidekick: Structuring scientifically sound surveys. In *Intelligent Tutoring Systems* (pp. 516-522). Springer International Publishing.

[28] Piech, C., Sahami, M., Koller, D., Cooper, S., & Blikstein, P. (2012, February). Modeling how students learn to program. In Proceedings of the 43rd ACM technical symposium on Computer Science Education (pp. 153-160). ACM

[29] Baum, L. E., Petrie, T., Soules, G., & Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The annals of mathematical statistics,* 41(1), 164-171.

Classifying behavior to elucidate elegant problem solving in an educational game

Laura Malkiewich
Teachers College, Columbia University
525 W 120th St.
New York, NY 10027
Laura.malkiewich@tc.columbia.edu

Ryan S. Baker
Teachers College, Columbia University
525 W 120th St.
New York, NY 10027
baker2@tc.columbia.edu

Valerie Shute
Florida State University
3205G Stone Building
1114 West Call St.
Tallahassee, FL 32306
vshute@fsu.edu

Shimin Kai
Teachers College, Columbia University
525 W 120th St.
New York, NY 10027
smk2184@tc.columbia.edu

Luc Paquette
University of Illinois, Urbana Champaign
383 Education Building
1310 S. Sixth St.
Champaign, IL 61820
lpaq@illinois.e

ABSTRACT

Educational games have become hugely popular, and educational data mining has been used to predict student performance in the context of these games. However, models built on student behavior in educational games rarely differentiate between the types of problem solving that students employ and fail to address how efficacious student problem solutions are in game environments. Furthermore, few papers assess how the features selected for classification models inform an understanding of how student behaviors predict student performance. In this paper, we discuss the creation and consideration of two models that predict if a student will develop an elegant problem solution (the Gold model), or a non-optimal but workable solution (the Silver model), in the context of an educational game. A pre-determined set of features were systematically tested and fit into one or both of these models. The two models were then examined to understand how the selected features elucidate our understanding of student problem solving at varying levels of sophistication. Results suggest that while gaming the system and lack of persistence indicate non-optimal completion of a problem, gaining experience with a problem predicts more elegant problem solving. Results also suggest that general student behaviors are better predictors of student performance than level-specific behaviors.

Keywords

Educational games; Problem solving, Classifiers.

1. INTRODUCTION

Educational games can be a great way to enhance learning; in some cases games lead to better learning than standard instructional activities [5, 22]. Yet while understanding how students learn in educational games is important, not much work has been done on modeling student learning in educational games that are open-ended, where students have a lot of freedom to explore. Furthermore, although there has been work on modeling behavior in games and educational learning environments to predict performance in these environments [6, 10, 13, 14, 16, 20] or more generally in school [4], there is not a lot of work that specifically looks at student problem solving strategies in games. Analyzing how students solve complex problems is a key part of understanding student learning in a domain [1, 3, 12], especially in open-ended environments [2]. For this reason, we are investigating student problem solving techniques in order to better understand the nature of student behavior and performance in open-ended educational games.

One key problem solving skill for learning is the ability to produce elegant solutions as well as workable solutions [8, 17], especially as one of the key markers of expertise in a field is the ability to solve problems more elegantly than a novice [11]. Even though there has been research on how to model different student approaches to problem solving [7] there has not yet been sufficient work on modeling the behaviors associated with elegant problem-solving vs. creating workable but less-optimal solutions to problems, especially in game environments. This paper examines how students solve problems to create elegant versus non-optimal, workable, solutions to problems in open-ended educational games. We study this issue in the context of Physics Playground, an open-ended discovery based learning game where students learn about Newtonian physics while trying to solve problems.

2. THE GAME: PHYSICS PLAYGROUND

Physics Playground, formerly called Newton's Playground [19], is an educational game that measures and supports knowledge of conceptual physics for middle and high school students. The game requires students to draw simple machines (consisting of ramps, levers, pendulums, and springboards) that act in accordance with Newton's laws of force and motion. In each level of the game, students are tasked with freehand drawing these machines, which are used to get a green ball to hit a red balloon. In addition to drawing machines, students can draw objects that interact with the ball directly in order to get the ball to reach the balloon. For example, students can draw objects made to fall and hit the ball directly, causing the ball to move. These objects are called "divers" in the context of the game. Students can also draw objects through the ball to move it up slightly. This technique is called "stacking" and is considered a form of "gaming the system" [21]. Similarly, students can click on the ball to "nudge" it forward slightly, if need be, without drawing an object at all. When students finally find a way to hit the red balloon with the green ball, they have completed the level, and are awarded a badge based on their performance.

Students can either receive a gold badge, silver badge, or no badge, depending on their performance in any given level. Badges are awarded according to the efficiency of the student's solution to a problem — determined by the number of objects a student draws in his or her attempt to solve a given problem. For most levels, gold badges are awarded if the student solves the problem by drawing three or fewer objects. Silver badges are awarded if the student solves the problem, but draws more objects. Each level is designed so that one simple machine (a ramp, springboard, pendulum or lever) will optimally solve the given problem. Accordingly, badges for performance are also tied to the type of machine that a student drew in the given level. For example, if a student creates an efficient solution to a level using a ramp, then

the student would be awarded a "gold ramp" badge upon completion of the level. Badges are awarded as a means to give students feedback about the efficiency of their solution, so students can reflect on their solution quality. Badges are not necessarily constructed for motivational purposes. Student badges are referred to as "trophies" in the context of the game, and are displayed in the top right hand side of the screen upon level completion.

The game consists of seven "playgrounds", or game worlds, that each contains 10-11 problems. In total there are 74 problems in the entire game. Problems are ordered by difficulty, and problem difficulty is determined by a number of factors including the location of the ball to the target, the magnitude and location of obstacles between the ball and the balloon, the number of agents required to get the ball to the balloon, the novelty of the problem. Students do not have to move through the game in a linear fashion. All levels are unlocked and accessible to students when the game starts (i.e., level access does not depend on a student's performance or progress in the game). Therefore, students can choose to go to any playground and work on any problem that they wish. That being said, there is a logical ordering to the levels, and many students do choose to go through the game in a linear fashion.

3. METHOD

3.1 The Study

This project is based on data collected during a prior study using Physics Playground. A more detailed description of the study population and methods can be found in [9, 18].

3.1.1 Participants

This data is from a study on 137 8th and 9th grade students who attended a diverse K-12 school in the southeastern United States.

3.1.1.1 Procedure

Students played the game in class for about 2.5 hours across four days of the study. Days 1 and 4 of the study consisted of student assessments, including a pretest and isomorphic posttest of students' knowledge of physics concepts. Learning data will not be discussed in the context of this paper [for learning data see 9, 18]. Days 2 and 3 of the study, as well as the first half of Day 4, consisted entirely of gameplay.

3.1.1.2 Measures

Physics Playground captured student log data during gameplay. The final data set consisted of 2,603,827 lines of action codes across the 137 students. Data collected included over seventy variables including information on student progression through the game, time stamps for actions, metrics on student drawings, gameplay actions, and badge awards. Across the 137 students, 919 levels were completed, 203 gold badges were awarded and 500 silver badges were awarded.

3.2 Model Selection

Two models were built for the purpose of distinguishing which features indicate elegant problem solving, and which indicate non-optimal problem solving. The first model was built to classify the award of a gold badge, where problem solutions are optimal (Gold Model). The second model was built to classify the award of a silver badge, where students solve a level, but in a non-optimal way (Silver Model). Levels that a student attempted but did not complete (levels where the student was not awarded a badge) were not used in this analysis.

By building two models, we were able to more effectively differentiate between features that predict elegant problem solving and features that predict non-optimal problem solutions more effectively. For example, creating two models allows for the identification of features that positively load onto one model but negatively load onto another. In turn, understanding these distinctions allows for a deeper understanding of how different levels of various features are indicative of the two types of problem solving. Badges were used as labels because they are the game's proxy for assessing student problem solution quality by marking the efficiency of a student's solution. Although badges in many modern games are used for motivational purposes, for the purpose of this project, we were only interested in what badges indicated about the elegance of a student's problem solution.

Features were created, tested, and iteratively improved upon, across a variety of classification algorithms. During this process, the J48 algorithm, which is Weka's implementation of the C4.5 algorithm [15], consistently provided the strongest predictive power, while protecting against over fitting. For this purpose, when it came to final feature selection and model creation, J48 was the sole algorithm used.

The models were built on less than half of the student data (61 students) so that the remaining test set could later be used to validate and test the final models. In order to validate the models during model creation and feature selection, batch level cross-validation was used. Each student was randomly assigned into 1 of 10 batches, and 10-fold validation was used to assess model goodness. Kappa was used as a measure of model fit.

3.3 Feature Selection

To make the two models, gold and silver labels were made. The gold label had a value of 1 if the student was awarded a gold badge, and a value of 0 if the student was awarded any other kind of badge (or no badge). A label for silver was created in the same way. The original log data tied each badge to the type of machine it awarded a badge for, but for the purpose of this project badge color and machine type were separated into two different features. This was done in part because we wanted to see if machine type affected which type of badge was awarded and in part because making machine type part of the label would result in models predicting what machine the student was building. Instead, we wanted to simply assess how successful students were at solving any given problem, regardless of the nature of the problem given.

Over fifty features were created and assessed for their goodness in predicting badge awards on any given level. The feature engineering process started with a restructuring of the raw student data logs to the problem-level (raw logs came at the action level) because the label of interest categorized student performance at the problem-level grain size. This process was then followed by a descriptive analysis of the variables that came out of this re-structured data, followed by structured brainstorming to elicit ideas about the types of features that could be built out of this data. Features were then created to measure certain constructs (e.g., time on task, gaming the system behavior, etc.) and behaviors of interest. Once a core set of features was created, colleagues and system experts were consulted about the quality, interest, and potential effectiveness of those features. Features were then iterated on. New features were created in an attempt to both measure constructs in more ways (e.g., measuring time on

task by looking at time on level, standardized time, or just time spent drawing objects) and to measure different student behaviors and constructs that the first set of features failed to measure. Features were then refined based on colleague and system expert feedback and used in single feature models to assess feature quality. An iterative process of feature creation, peer consulting, and feature refinement then continued for several more cycles until the final set of fifty features had been created.

Once all features had been created, single-feature models were used to choose the seventeen features that were the best predictors of any given construct. For example, Time on Level in minutes was determined to be a better classification of the amount of time that a student spent on a level than standardized time.

The final seventeen features were then ordered in terms of their goodness within a single-feature J48 model, under student-level cross-validation. The best feature was added, and then a recursive process was used where additional features were tested in the same order to determine whether adding that feature improved model goodness, as measured by an increase in kappa. Only features that improved kappa were added. The final gold model contained fourteen features, and the final silver model contained nine features.

3.4 Feature Descriptions

The final seventeen features used for model creation are listed and described below in addition to which model they ended up being included in. Features are listed in the order that they were tested and selected.

Sum Elapsed (silver): The total amount of time that a student spent actively drawing objects up until that point in the game. For example, if a student spends 90 seconds actively drawing objects in Level 1, and then 30 seconds drawing during Level 2, then Sum Elapsed by the end of Level 2 would have a value of 120 seconds.

Time on Level (both): The total amount of time spent playing the level that the student is being awarded the badge for (in seconds).

Nudge Count (gold): The total number of times that the student pressed the ball to nudge it forward a little in the level.

Number of Objects (both): The total number of distinct objects (machines, random lines, weights, etc.) the student drew in the level.

Diver Count (none): The total number of divers that a student created in the level.

Pause Before End (both): Binary indicator of whether or not the student hit the pause button as their last action before the level ended. Usually this happens when students wants to exit out of a level before completing the level. In this case, students would neither be awarded a gold badge nor a silver badge.

Ball Count (both): The number of balls a student uses in a level. If a student knocks a ball off the screen or if the ball provided to the student falls to the bottom of the screen, then it disappears and the student gets a new ball to try again.

Max Velocity Y (both): The maximum velocity that any ball a student used in a level traveled in the y direction (up and down). Velocity values in the Physics Playground system are given in meters-kilogram-second (MKS) units.

Max Velocity X (gold): The maximum velocity that any ball a student used in a level ever traveled in the x direction (left and right).

Erased Object Count (silver): Number of objects that a student drew, and then erased in the level. Students can erase an object that they have drawn by clicking on it.

Stack Count (both): Number of times student drew an object through the ball in order to move the ball up.

Badge Before (gold): Binary indicator of whether or not a student has received a badge (of any color) on this level before.

Played Before (gold): Binary indicator of whether or not a student has played this level before.

Average Free-fall Distance (gold): Free-fall distance is a measure of how far any divers fell before striking a ball. This feature averages across all those distances in the level. Units are percentage of the game screen. So if the diver falls half the distance of the game screen, this would have a value of 0.5.

Restart Count (gold): The number of times a student re-started the level.

Play Count (gold): The number of times that a student has played the current level before. Restarts are not included in this count. A student has to have either completed the level or made an attempt at the level, left the level, and then returned, in order for it to contribute towards this play count.

Machine (both): The type of machine that should be created to optimize movement of the ball to the target. There is one machine per level and they can take the form ramp, lever, pendulum, or springboard.

3.5 Final Models

The final J48 gold classification model with ten-fold student batch cross-validation, which was built on half the data, had a Kappa value of 0.69, and the silver classification model had a Kappa of 0.83. The other half of the data was held out for future analysis comparing the models developed here to other, future models. As is evident from the features mentioned above, seven features fit into both the gold and silver classification models. Those features were Time on Level, Number of Objects, Pause Before End, Ball Count, Max Velocity Y, Stack Number, and Machine. Seven features only fit the gold classification model; those were Nudge Count, Max Velocity X, Badge Before, Played Before, Average Free-fall Distance, Restart Count, and Play Count. Finally, two features only fit the silver classification model. Those were Sum Elapsed and Erased Object Count.

3.6 Qualitative Analysis of Models

The primary goal of this project was to use classification models to help elucidate how student behavior predicts gold and silver badge acquisition differently. For this reason, we take a more qualitative look at which features were included in each model, which were included in both, and which were included in neither.

Table 1 indicates how each of the features loaded onto each of the models when used in a single-feature model (machine type does not have a numeric value, so it is not included in the table). Since both models were built using J48 decision trees, this is simply a proxy for the general loading of each feature on the model outcomes, and not a comprehensive measure of how each feature fits into each model.

Table 1. Feature loadings onto each model

Feature	Gold Model	Silver Model
Sum Elapsed	-	Negative
Time on Level	Negative	***Positive***
Nudge Count	Negative	-
Number of Objects	Negative	***Positive***
Diver Count	-	-
Pause Before End	Negative	Negative
Ball Count	***Positive***	Negative
Max Velocity Y	Negative	***Positive***
Max Velocity X	***Positive***	-
Erased Object Count	-	***Positive***
Stack Count	Negative	***Positive***
Badge Before	Negative	-
Played Before	Negative	-
Average Free-fall Distance	Negative	-
Restart Count	***Positive***	-
Play Count	***Positive***	-

3.6.1 Features included in both models

Features that were included in both models mostly helped indicate whether the student was able to achieve optimal performance or simply workable solutions. For the majority of the features that were in both models, the value was higher for non-gold and higher for silver, indicating that these behaviors were typical of students who developed workable yet non-optimal solutions.

For example, Time on Level was a good indicator of which students produced non-optimal, yet workable solutions. Students who spent a very short time on the level could have entered a level and then immediately quit, so they were likely to not receive a badge. However, longer time in level is associated with a badge but not a gold badge. This loading is likely because students who spend a long time on a level are struggling more or drawing more and those students are therefore less likely to develop the most optimal solution in a single level attempt.

Other features that were higher for non-gold and silver were Number of Objects, Max Velocity Y, and Stack Count. It makes sense that students who drew more objects would get silver, because they are doing more work than students who quit the level (no badge) and students who developed optimal solutions (gold badge). Also, badges are awarded in accordance with the number of objects a student draws in his or her attempt to solve a given problem, so it makes sense that this feature would be a significant indicator of performance. Stack Count could have been a good indicator of whether students solved a problem optimally or non-optimally because students who are stacking a lot could be trying to game the system, likely because they don't know how to solve the problem more effectively using machines. These students are likely to get a silver badge if they complete the problem, because stacking requires drawing many objects.

Only one feature that appeared in both models was higher for both non-gold and non-silver, Pause Before End. This is likely because students who paused before the end of the level were quitting, and therefore did not receive a badge at all. However, that was not always the case.

It is curious that students who had a higher Ball Count per level were more likely to produce optimal solutions; the value for ball count was higher for gold and non-silver indicators. This may be because students who created optimal solutions were experimenting more, and therefore going through more balls, but without spending too much time or drawing too many objects. This behavior could be indicative of students who are quickly iterating on a single idea, or thinking of what to do before drawing objects. (On some levels balls keep dropping down until you draw an object underneath to catch the ball, so the longer you spend without drawing an object, the more balls you use).

3.6.2 Features that only fit the gold model

Features that only fit the gold model are interesting because they specifically separate those who were able to solve problems elegantly as opposed to students who could not find an optimal solution to the problem. The features fit three general categories, relative to whether or not they indicate experience, shallow strategies, or efficiency.

Features that indicate experience include Badge Before, Played Before, Play Count, and Restart Count. It is interesting that Badge Before and Played Before, which are both binaries, indicate non-Gold performance while Play Count and Restart Count indicate gold performance. This indicates that if a student is working on a problem they have completed or played once before, they are not likely to develop an optimal solution, but the more they play a level, the closer they are to get to an optimal solution. Students who have played the level before have some experience with the problem space, even if they did not complete the level previously and that experience could help them determine an optimal problem solution. Play Count and Restart Count tell the model the precise amount of experience the current student has had with a level. Students who re-start or play a level more often might be optimizers, aiming to iterate several times on their problem solution in an attempt to find the best approach to solving the problem. They might be thinking more critically about the choices they are making and choosing to come back to a level or start it again when they've determined that they have acquired the skill or knowledge necessary to now perform more effectively. Resetting also enables students to clear their screens of all objects, and start over, so they can approach the problem afresh. This can be a good strategy for students who want to try going in a different direction instead of iterating on an earlier idea, and it can lead to more efficient problem attempts later.

Nudge Count is a feature that indicates shallow strategies, or even potentially gaming the system. Students who nudge the ball a lot are trying to make the ball move without using a drawn machine to move the ball. This could lead to effectively moving the ball without drawing more objects, which could lead to a problem solution despite a low object count, which would result in a gold badge. Or, it could indicate a student who is nudging because they are struggling a lot with the problem, perhaps because they have

already drawn many objects, but are unable to get the ball to move effectively, so they try to nudge it along.

The other features associated with gold badges but not silver badges measure how efficiently students are building machines. These include Average Free-fall Distance and Max Velocity X. Max Velocity X is a predictor of gold badges while Max Velocity Y can predict gold and silver badges, because Max Velocity X is a more effective measure of how well a student has constructed his or her machine. If a ball is dropped from the starting point, then regardless of how effective the student's machine is, the ball will, in many cases, hit the same maximum velocity as it falls because all balls in the Physics Playground interface follow the laws of physics, and therefore accelerate at g. However, how fast a ball moves in the x direction is a direct result of how well a student's designed machine moved the ball in that direction. Likewise, Average Free-fall distance measures student machine efficiency, because students have to carefully choose where to draw divers so that they have a desired effect on ball movement. Divers that are positioned too far away might not hit the desired target, requiring another driver to be drawn for the desired effect. Therefore, both these features are found in this model because they are able to successfully classify effective and efficient student construction choices.

3.6.3 Features that only fit the silver model

Only two features were associated with silver badges but not gold badges. They were Sum Elapsed and Erased Object Count. Both of these features describe the behaviors of students who are tinkering to iterate to a solution. Sum Elapsed negatively loads on the model, suggesting that it indicates ineffective tinkering, while Erased Object count positively loads on the model, suggesting that it indicates effective yet inefficient tinkering. Sum Elapsed is a measure of how much effort a student has put into the game, up until that point in time. A student who has spent a lot of time drawing objects across all prior game levels will have a higher Sum Elapsed value. This is higher for non-silver badges, maybe in part because students who spend a lot of time drawing on levels are less likely to complete the level they are on. This could be because students are making long strokes while doodling, or doing other off task work. On the other hand, students who erase many objects are more likely to get a silver badge. This might be because students who erase a lot are pruning their work if they drew too many objects or made mistakes. These students are more dedicated to completing the current problem, to acquire a badge, but they are not likely to solve the problem in an optimal manner. Therefore Erased Object Count measures an effective problem solving strategy that is not efficient.

3.6.4 Features that fit neither model

It is important to consider not only the features that fit into the models, but also the features that failed to improve either of the models when added. These included Diver Count and a host of other features that were discarded during the feature engineering process, due to the features' low predictive power for behaviors of interest. Interestingly, more specific features involving specific machines or operators were less predictive of student performance than more general variables. Concrete behavior-specific features like Diver Count and Pin Count (pins are small dots that students can add to a drawing to tack an object in place or create a point for an object to rotate around) were less associated with outcomes than were general features like Object Count and Sum Elapsed, which describe student behaviors that span across several actions or several levels. (Note that divers are objects, so when talking about a distinction between these features Object Count is a more general category than Diver Count). It could be that student performance on any particular problem was not as predictive of their problem-solving efficacy as that student's overall behavior. This could suggest that problem solving scaffolding and teaching should focus more on students' overall strategies, rather than level specific strategies. On the other hand, it may simply indicate that none of the more specific features, by themselves, are as predictive as the more general categories that cut across and combine different specific features. It is also important to note that in addition to improving prediction, using more general features also reduces the risk of models over-fitting.

4. DISCUSSION AND CONCLUSION

This analysis of two models built to predict optimal student performance and non-optimal student performance gives us some interesting insights about the kinds of behaviors that predict student performance, and also about the kinds of features that best fit these types of models. Models that describe student performance more generally are more predictive when fed into a J48 decision tree, which can make cutoffs at different values of those feature variables in order to differentiate students who are solving levels optimally, sub-optimally, and not solving levels at all. In turn, features that differentiate optimal performers from all others focus on student experience with the problem space, shallow strategies, and gaming behaviors in addition to measures of student problem solving efficiency. Classifiers of successful but sub-optimal performance tend to describe more exploratory, tinkering behavior while classifiers of elegant problem solving seem to highlight the value of student exposure to a problem and measures of problem-solving efficiency.

These findings give insight into future designs of Physics Playground and other games and open-ended learning environments. To encourage more elegant student problem solving, the learning environment can encourage students to revisit problems, especially after they've created a workable solution, but failed to create an elegant one. Additionally, student feedback about how effective their solution is or what kind of metrics are needed for an optimal solution (e.g., a prompt indicating that for the ball to reach the target it must hit a certain x velocity) could aid students in understanding what more proximal goals they need to fulfill in order to ultimately solve the problem at hand in the most efficient way.

Future work can explore whether similar features are effective for predicting student problem solving in other games. The models discussed here were built on only one game with a unique form of gameplay and specific design constraints, so the study is limited in its generalizability. However, there is the potential for the results of this paper to be used for constructing models for classifying student performance to differentiate between elegant and non-optimal problem solving strategies in other games or open-ended learning environments.

5. ACKNOWLEDGMENTS

We would like to thank the Bill & Melinda Gates Foundation (SOL1071343/APP181499) for their generous support provided for this research, and Ed Dieterle for helpful suggestions and advice. We would also like to thank the students who participated in the study.

6. REFERENCES

[1] Anderson, J. R. (1993). Problem solving and learning. *American Psychologist*, *48*(1), 35.

[2] Blikstein, P. (2011, February). Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st international conference on learning analytics and knowledge* (pp. 110-116). ACM.

[3] Chi, M. T., Feltovich, P. J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, *5*(2), 121-152.

[4] Chi, M., Schwartz, D. L., Chin, D. B., & Blair, K. P. (2014, July). Choice-based Assessment: Can Choices Made in Digital Games Predict 6 th-Grade Students' Math Test Scores?. In *Proceedings of the 7th International Conference on Educational Data Mining (EDM 2014)*(pp. 36-43).

[5] Clark, D. B., Tanner-Smith, E. E., & May, S. K. (2013). *Digital games for learning: A systematic review and meta-analysis*.

[6] Conrad, S., Clarke-Midura, J., & Klopfer, E. (2014). A framework for structuring learning assessment in a massively multiplayer online educational game: experiment centered design, *International Journal of Game Based Learning, 4(1)*, 37-59.

[7] Eagle, M., & Barnes, T. (2014, July). Exploring differences in problem solving with data-driven approach maps. In *Proceedings of the 7th International Conference on Educational Data Mining (EDM 2014)*(pp. 76-83).

[8] Ertmer, P. A. (2015). *Essential Readings in Problem-based Learning*. Purdue University Press.

[9] Kai, S., Paquette, L., Baker, Bosch, N., D'mello, S., Ocumpaugh, J., Shute, V., & Ventura, M. (2015). A Comparison of face-based and interaction-based affect detectors in physics playground. In *Proceedings of the 8th International Conference on Educational Data Mining (EDM 2015)*(pp. 77-84).

[10] Koedinger, K. R., Stamper, J. C., McLaughlin, E. A., & Nixon, T. (2013, July). Using data-driven discovery of better student models to improve student learning. In *Artificial intelligence in education* (pp. 421-430). Springer Berlin Heidelberg.

[11] Larkin, J., McDermott, J., Simon, D. P., & Simon, H. A. (1980). Expert and novice performance in solving physics problems. *Science*, *208*(4450), 1335-1342.

[12] Larkin‡, J. H., & Reif, F. (1979). Understanding and teaching problem-solving in physics. *European Journal of Science Education*, *1*(2), 191-203.

[13] Martin, J., & VanLehn, K. (1995). Student assessment using Bayesian nets.*International Journal of Human-Computer Studies*, *42*(6), 575-591.

[14] Olsen, J. K., Aleven, V., & Rummel, N. Predicting Student Performance In a Collaborative Learning Environment. In *Proceedings of the 8th International Conference on Educational Data Mining (EDM 2015)*(pp. 211-217).

[15] Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufman: New York.

[16] Rowe, E., Baker, R., Asbell-Clarke, J., Kasman, E., & Hawkins, W. (2014, July). Building automated detectors of gameplay strategies to measure implicit science learning. In *Poster presented at the 7th annual meeting of the international educational data mining society* (pp. 4-8).

[17] Savransky, S. D. (2000). *Engineering of creativity: Introduction to TRIZ methodology of inventive problem solving*. CRC Press.

[18] Shute, V.J., D'Mello, S., Baker, R.S.J., Cho, K., Bosch, N., Ocumpaugh, J., Ventura, M., & Almeda, V. (2015). Modeling how incoming knowledge, persistence, affective states, and in-game progress influence student learning from an educational game. *Computers & Education, 86*, 224-235.

[19] Shute, V.J., Ventura, M., & Kim, Y.J. (2013). Assessment and learning of informal physics in Newton's Playground. *The Journal of Educational Research, 106*, 423-430.

[20] VanLehn, K. (1988). Student modeling. *Foundations of intelligent tutoring systems*, *55*, 78.

[21] Wang, L., Kim, Y. J., & Shute, V. (2013). "Gaming the system" in Newton's Playground. In *AIED 2013 Workshops Proceedings Volume 2 Scaffolding in Open-Ended Learning Environments (OELEs)* (p. 85).

[22] Wouters, P., Van Nimwegen, C., Van Oostendorp, H., & Van Der Spek, E. D. (2013). A meta-analysis of the cognitive and motivational effects of serious games. *Journal of Educational Psychology*, *105*(2), 249.

Predicting Dialogue Acts for Intelligent Virtual Agents with Multimodal Student Interaction Data

Wookhee Min
North Carolina State University
Raleigh, NC 27695
wmin@ncsu.edu

Joseph B. Wiggins
North Carolina State University
Raleigh, NC 27695
jbwiggi3@ncsu.edu

Lydia G. Pezzullo
Tufts University
Medford, MA 02155
lydia@learndialogue.org

Alexandria K. Vail
North Carolina State University
Raleigh, NC 27695
akvail@ncsu.edu

Kristy Elizabeth Boyer
University of Florida
Gainesville, FL 32611
keboyer@ufl.edu

Bradford W. Mott
North Carolina State University
Raleigh, NC 27695
bwmott@ncsu.edu

Megan H. Frankosky
North Carolina State University
Raleigh, NC 27695
rmhardy@ncsu.edu

Eric N. Wiebe
North Carolina State University
Raleigh, NC 27695
wiebe@ncsu.edu

James C. Lester
North Carolina State University
Raleigh, NC 27695
lester@ncsu.edu

ABSTRACT

Recent years have seen a growing interest in intelligent game-based learning environments featuring virtual agents. A key challenge posed by incorporating virtual agents in game-based learning environments is dynamically determining the dialogue moves they should make in order to best support students' problem solving. This paper presents a data-driven modeling approach that uses a Wizard-of-Oz framework to predict human wizards' dialogue acts based on a sequence of multimodal data streams of student interactions with a game-based learning environment. To effectively deal with multiple, parallel sequential data streams, this paper investigates two sequence-labeling techniques: long short-term memory networks (LSTMs) and conditional random fields. We train predictive models utilizing data corpora collected from two Wizard-of-Oz experiments in which a human wizard played the role of the virtual agent unbeknownst to the student. Empirical results suggest that LSTMs that utilize game trace logs and facial action units achieve the highest predictive accuracy. This work can inform the design of intelligent virtual agents that leverage rich multimodal student interaction data in game-based learning environments.

Keywords

Game-Based Learning, Virtual Agents, Deep Learning, Multimodal.

1. INTRODUCTION

Recent years have witnessed a growing interest in intelligent game-based learning environments because of their potential to simultaneously promote student learning and create engaging learning experiences [23]. These environments incorporate personalized pedagogical functionalities delivered with adaptive learning techniques and the motivational affordances of digital games featuring believable characters and interactive story scenarios situated in meaningful contexts [13, 23]. A key feature of game-based learning environments is their ability to embed problem-solving challenges within interactive virtual environments, which can enhance students' engagement and facilitate learning through customized narratives, feedback, and problem-solving support [18, 25].

Game-based learning environments offer considerable opportunities for implementing virtual agents by delivering visually contextualized pedagogical strategies [14]. Intelligent virtual agents have been shown to deliver motivational benefits, promote problem-solving, and positively affect students' perception of learning experiences [14]. Virtual agents play a variety of roles in interactive learning environments including intelligent tutors, teachable agents, and learning companions [4].

A key challenge in developing intelligent virtual agents is devising accurate predictive models that dynamically attune pedagogical strategies to individual students using evidence from students' interactions with the learning environment. Previous research has focused on when to intervene [21] and what types of dialogue moves to make during students' problem-solving activities [3] to provide support in a timely, contextually relevant manner. Selecting appropriate pedagogical dialogue moves is critical [24] because failing to provide effective feedback may lead to decreased learning in a student experiencing boredom [1], lead a student who is confused to become disengaged [10], or negatively impact the outcome of dialogues [5].

Much of the previous work in this line of investigation has addressed this challenge through computationally modeling agents' *dialogue acts*, the underlying intention (e.g., greeting, question, suggestion) of the utterances, by utilizing sequences of actions within learning environments as evidence [2]. The current work builds on this by examining multimodal data streams, which

can provide rich evidence of students' cognitive and affective states, in addition to evidence captured from game trace logs. To effectively deal with the granular sequential data in parallel multimodal data streams, we investigate two sequence labeling techniques: a deep-learning technique, long short-term memory networks (LSTMs) [11]; and a competitive baseline approach, conditional random fields (CRFs) [26]. This work is inspired by the recent success of LSTMs in dealing with low-level data (e.g., speech signals), and particularly by their state-of-the-art performance in speech recognition tasks [16]. Additionally, hierarchical representation learning supported by deep learning provides advantages over other machine learning techniques by avoiding the need for labor-intensive feature engineering [16].

Our sequence labeling models are evaluated with 211 dialogue acts made by human wizards who interacted with 11 students playing CRYSTAL ISLAND, a game-based learning environment for middle school microbiology [23]. The interaction data include game trace logs, facial action units [17] processed from facial video recordings, and galvanic skin responses, all of which are utilized as input features for devising predictive models. Wizards used pre-designed utterances, which they selected from menus organized by dialogue act. Each selected utterance was then delivered to the student via speech synthesis. Wizards could observe the student's face, gaze, game screen, and voice while selecting dialogue moves, but facial action units, galvanic skin responses, and game trace logs were not directly accessible. We hypothesize that these unobserved multimodal data streams serve as proxies for the wizards' dialogue decisions and examine these as explanatory variables to predict the next dialogue act that a human wizard might choose.

LSTM and CRF models are devised utilizing subsets of the parallel multimodal data streams. Student-level cross-validation studies indicate that LSTMs utilizing game trace logs and facial action units outperform both CRFs and the majority class-based baseline with respect to predictive accuracy. Further, we find that the LSTM model effectively takes advantage of multimodal data streams, and it most effectively utilizes both game trace logs and facial action unit data. The results suggest that LSTM models can serve as the foundation for dialogue act modeling for intelligent virtual agents that dynamically adapts dialogues to individual students.

2. RELATED WORK

Recent work in game-based learning has explored a broad spectrum of subject matters ranging from computer science [18] and language to cultural learning [13]. Narrative-centered learning environments, which provide narrative adaptation for individual students in the context of intelligent game-based learning, have been found to deliver experiences in which learning and engagement are synergistic [13, 23]. Student interaction data from game-based learning activities has provided a rich source of information from which students' development of competencies [18, 25] and progress towards learning goals [19, 20] are diagnosed. Game-based learning environments can also be populated by virtual agents, whose design should consider students' cognitive and affective states [4, 14].

In parallel work on tutorial dialogue, it has been found that tutorial planning can take into account students' cognitive and affective states [7]. Planning dialogue moves and inducing turn-taking policies have been widely examined in supervised learning (e.g., hidden Markov models [2], directed graph representations [5]) and reinforcement learning [3, 21]. The approach described in this paper is the first to investigate dialogue move classification using LSTMs and CRFs that take as input sequential multimodal data streams, which can serve as the foundation for guiding the dialogue of intelligent virtual agents in game-based learning environments.

Figure 1. The CRYSTAL ISLAND game-based learning environment.

3. CRYSTAL ISLAND

Over the past several years, our lab has been developing CRYSTAL ISLAND (Figure 1), a game-based learning environment for middle school microbiology [23]. Designed as a supplement to classroom science instruction, CRYSTAL ISLAND's curricular focus has been expanded to include literacy education based on Common Core State Standards for reading informational texts. The narrative focuses on a mysterious illness afflicting a research team on a remote island. Students play the role of a visitor who is drawn into a mission to save the team from the outbreak. Students explore the research camp from a first-person viewpoint, gather information about patient symptoms and relevant diseases, form hypotheses about the infection and its transmission source, use virtual lab equipment and a diagnosis worksheet to record their findings, and report their conclusions to the camp's nurse.

Extending the previous edition of CRYSTAL ISLAND, we incorporated a prototype virtual agent into the game to investigate both affective and cognitive influences on students' learning processes. This virtual agent, a young female scientist named Layla (Figure 2), was designed as a near-peer mentor who supports the student through dialogue-based interactions.

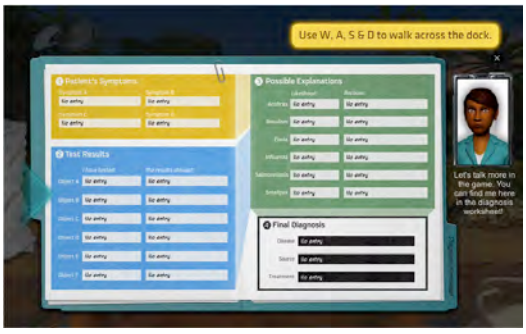


Figure 2. CRYSTAL ISLAND virtual agent.

In CRYSTAL ISLAND'S virtual world, students interact with learning resources such as books and posters, as well as with non-player characters through informative menu-based dialogue. As students progress through the game, they collect evidence and record their hypotheses in a "diagnosis worksheet." The student meets Layla when the diagnosis worksheet is opened (Figure 2).

With Layla's visual and speech synthesis prototypes in place, but no adaptive dialogue model implemented yet, a Wizard of Oz system was implemented to enable a human operator to provide the intelligence behind Layla's dialogue. When the human "wizard" decides to initiate a dialogue move, she chooses one of six dialogue acts (Table 1) from a menu interface, then selects a dialogue utterance from the act's set of pre-determined utterances. Layla then speaks the utterance through speech synthesis. The selection of dialogue moves was informed by the literature on dialogue systems for learning [8], as well as experience with a recent study conducted in the same middle school, in which pairs of middle school students interacted with CRYSTAL ISLAND together.

Three wizards controlled Layla's dialogue in the game from a room separated from the students, while observing the students through a live feed that included the student's facial video, the student's gaze superimposed in real time over a video capture of the game screen, and the student's voice as recorded through a headset microphone.

Data was collected in two studies implemented in the spring and summer of 2015 at a public middle school in Raleigh, North Carolina. In the spring study, participants were drawn from an after-school activity, and the summer study's participants were from classroom pull-outs. Of the 11 students who participated, 7 were female and 4 were male, with an average age of 12 (SD = 1.1). The data corpus contains 211 virtual agent dialogue acts across the students (average number of acts: 19.2, maximum number of acts: 41, and minimum number of acts: 3).

Table 1. Agent's dialogue acts and distributions of their use.

Dialogue Act	Distributions	Dialogue Act	Distributions
Greeting	58 (27.5%)	***Suggestion***	51 (24.2%)
Question	35 (16.6%)	***Feedback***	8 (3.8%)
Acknowledge-ment	43 (20.4%)	***Affective Statement***	16 (7.6%)

4. MULTIMODAL DATA

During the students' interactions with CRYSTAL ISLAND, both game actions and parallel sensor data were captured to collect both cognitive and affective features of students' experience. In the following subsections, we describe the three types of input data investigated in the present work.

4.1 Game Trace Logs

Students play CRYSTAL ISLAND using a keyboard and mouse. Student actions are logged for gameplay analysis and game telemetry [20]. In the present modeling work, seven key categories of actions are examined: moving around the camp, using the laboratory's equipment to test a hypothesis about the disease and its source, conversing with non-player characters, reading complex informational texts about microbiology concepts, taking embedded assessments associated with the informational texts, interacting with the diagnosis worksheet, and experiencing dialogue moves with the virtual agent. The total number of distinct actions is 143.

A total of 4,117 student actions were logged along with 211 dialogue acts by the virtual agent in the training data. Students took an average of 19.5 actions between two adjacent dialogue acts, where the minimum and maximum number of actions between any two adjacent dialogue acts are 1 and 217, respectively.

4.2 Galvanic Skin Response

Galvanic skin response (GSR) is a measurement of the level of conductance across the surface of the skin, which is driven by the activity of the sympathetic nervous system. GSR reflects a variety of cognitive and affective processes, including attention and engagement [6, 22]. In addition, the presence of significant spikes in students' GSR in response to certain events during a technology-supported learning activity has been found to be associated with learning-linked emotions and learning outcomes [12]. In this study, Empatica E4 bracelets on both wrists were used for GSR recording. These bracelets were chosen because, unlike palmar and fingertip GSR recording devices, they do not restrict the range of hand movement needed to play the game.

4.3 Facial Action Units

Facial expressions have been shown to have a relationship to self-reported and judged learning-centered affective states [1, 17]. Previous work has also found that facial expressions during learning can help predict a student's learning gains, frustration, and engagement [27]. Facial expressions can be examined non-invasively through video recordings taken during a student's interaction with a learning environment.

In this work, we observe facial expressions by analyzing a student's facial action units, which capture movement of the muscles in the face. Facial action units are grounded in the Facial Action Coding System, which was devised to make observations about facial movements [9]. In this study, facial videos were recorded via a webcam and analyzed using FACET, an automated system devised for tracking facial action units, because it allows for frame-by-frame tracking in the facial videos without the time intensive effort of human-tagging facial action units. FACET is the next generation of the Computer Expression Recognition Toolbox [17], which has been validated for both adults and children. In this study, we considered the subset of facial action units provided by FACET (Table 2). In the following section, we describe the deep learning-based dialogue act classifier that utilizes these three data sources.

Table 2. Facial action units examined.

Inner Brow Raiser (AU1)	Upper Lip Raiser (AU10)	Tightener (AU23)
Outer Brow Raiser (AU2)	Lip Corner Puller (AU12)	Lip Pressor (AU24)
Brow Lowerer (AU4)	Dimpler (AU14)	Lips Part (AU25)
Upper Lid Raiser (AU5)	Lip Corner Depressor (AU15)	Jaw Droop (AU26)
Cheek Raiser (AU6)	Chin Raiser (AU17)	Lip Suck (AU28)
Lid Tightener (AU7)	Puckerer (AU18)	
Nose Wrinkler (AU9)	Lip Stretcher (AU20)	

5. LSTM-BASED DIALOGUE MOVE DECISION MODEL

Long short-term memory networks (LSTMs) have demonstrated significant success in dealing with a series of raw signals, such as speech, yielding state-of-the-art performance in speech recognition tasks [16]. This inspires our work, which deals with low-level sensor data such as GSRs and facial AUs. In the following subsections, we present a high-level description of LSTMs [11], introduce how multimodal input data are synchronized and encoded into a trainable format, and describe how the LSTM-based dialogue move prediction models are configured.

5.1 LSTM Background

LSTMs are a type of gated recurrent neural network specifically designed for sequence labeling on temporal data. LSTMs, like standard recurrent neural networks, take the approach of sharing weights across layers at different time steps. LSTMs feature a sequence of memory blocks that include one or more self-connected memory cells along with three gating units [11]. In LSTMs, the input and output gates modulate the incoming and outgoing signals to the memory cell, and the forget gate controls whether the previous state of the memory cell is remembered or forgotten. This structure allows the model to preserve gradient information over longer periods of time [11].

In the implementation of LSTMs investigated here, the input gate (i_t), forget gate (f_t), and candidate memory cell state ($\widetilde{c_t}$) at time *t* are computed by Equations (1)–(3), respectively, in which *W* and *U* are weight matrices for the input (x_t) at time *t* and the cell output (h_{t-1}) at time *t*-1, *b* is the bias vector of each unit, and σ and *tanh* are the logistic sigmoid and hyperbolic tangent function, respectively.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (2)$$

$$\widetilde{c_t} = tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3)$$

Once these three vectors are computed, the current memory cell's state is updated to a new state (c_t) by modulating the current memory cell state candidate value ($\widetilde{c_t}$) via the input gate (i_t) and the previous memory cell state (c_{t-1}) via the forget gate (f_t). Through this process, a memory block decides whether to keep or forget the previous memory state and regulates the candidate of the current memory state via the input gate. This step is described in Equation (4), in which $\odot$ denotes element-wise multiplication:

$$c_t = i_t \odot \widetilde{c_t} + f_t \odot c_{t-1} \quad (4)$$

The output gate (o_t) calculated in Equation (5) is utilized to compute the memory cell output (h_t) of the LSTM memory block at time *t*, modulating the updated cell state (c_t) (Equation 6):

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (5)$$

$$h_t = o_t \odot tanh(c_t) \quad (6)$$

Once the cell output (h_t) is calculated at time *t*, the next step is to use the computed cell output vectors to predict the label of the current training example. For the dialogue move decision model, we use the final cell output vector (h_t), assuming that h_t captures long-term dependencies from the previous time steps.

5.2 Data Encoding for Dialogue Move Decision Model

Each data stream from a suite of multimodal interaction data is of a sequential form. Because these data include fixed-rate recordings (e.g., facial action units and galvanic skin responses) with rates that differ between streams, as well as in-game action-driven recordings (e.g., game trace logs) with no set rate, the first step of data encoding is synchronizing input data across modalities.

We obtained from each student two series of galvanic skin responses (GSRs), one each for the left and right hand, as well as 19 facial action units (AUs). In the modeling work reported here, only the GSR information from the subject's dominant hand is utilized, so GSR is represented by a one-dimensional vector. AUs are represented by a 19-dimensional vector space per time stamp. GSR and AUs were logged with the frequencies of approximately 4Hz and 30 Hz, respectively. Game traces were recorded as events were triggered in the game, whenever the actions described in Section 4.1 were performed.

In contrast to GSR or AUs, which have continuous values, the game trace logs (GAME) consist of discrete indices for specific actions, indexed 1 to 143. To represent actions in a vector format, we employ the *one-hot-encoding* technique, in which a bit vector whose length is the total number of actions (143 in this work) is created while only the associated action bit is on (i.e., 1) while all other bits are off (i.e., 0). Once the vector representations for GAMEs are created, the next step is to synchronize the three data representations into an integrated representation.

To keep the length of data sequences manageable while preserving key game actions, we synchronize the multimodal data based on the game trace logs. All GSR and AU data collected between any two adjacent game actions are transformed into two vectors, using the following method:

- Vector 1: (75th percentile minus 50th percentile) per feature across all the data points between the two adjacent actions
- Vector 2: (50th percentile minus 25th percentile) per feature across all the data points between the two adjacent actions

We hypothesize that these two quartile-based vectors can capture variance of signals within an interval, while effectively avoiding outliers, smoothing out individual differences, and keeping the number of input features (183, or the sum of 143 for GAME, 38 for AU, and 2 for GSR) small enough to efficiently train LSTMs. Once these two vectors are created for the GSR stream and for each AU, the vectors are concatenated to the game trace log vector.

5.3 LSTM Model Configurations for Dialogue Move Decision

Prior to training LSTMs, the hyperparameters of the models must be determined. LSTM hyperparameters have often been explored using grid search or random search settings in the process of minimizing validation errors [20]. We adopt the grid search approach to empirically find an optimal configuration for a set of hyperparameters. In this work, we consider two hyperparameters: the number of hidden units for LSTMs among {32, 64} and the dropout rate [16], a model regularization technique, among {0.4, 0.7}. Both hyperparameters have significant influence on the performance of deep neural networks [11, 20].

In addition to LSTM-wide hyperparameters, this work also analyzes the isolated impacts of multimodal data sources. In order to perform this analysis, we examine all possible combinations of features, generating the following seven input feature sets: galvanic skin responses (GSRs), facial action units (AUs), game trace logs (GAMEs), GSRs and AUs, AUs and GAMEs, GSRs and GAMEs, and all three data sources. The dimension of a feature set is decided by summing up the dimensions of the features (see Section 5.2) that comprise the feature set.

In addition to the hyperparameters examined in the grid search, we apply a fixed value to the following hyperparameters for LSTMs: employing a softmax layer for classifying given sequences of interactions, adopting mini-batch gradient descent with a mini-batch size of 32, utilizing categorical cross entropy for the loss function, and employing a stochastic optimization method. The training process stops early if the validation score has not improved within the last 15 epochs. In this work, we evaluate our models using student-level leave-one-out cross validation, and so in each fold, 1 student's data is used for testing

(completely hidden) out of 11 students, while 8 students' and 2 students' data are utilized as the training and validation set, respectively. Finally, the maximum number of epochs is set to 100.

6. EVALUATION

To evaluate the proposed LSTM-based dialogue act classification (cast as six-class classification), we search for an optimal set of hyperparameters through cross-validation in the previously discussed grid search setting, and then perform feature-set level predictive performance analyses based on the chosen hyperparameters. Additionally, we compare each LSTM-based computational model to a competitive approach based on linear-chain conditional random fields (CRFs) [26] as well as a majority class baseline using the same cross-validation split for a pairwise comparison. CRFs are trained using the Block-Coordinate Frank-Wolfe optimization technique [15], and we adjust the regularization parameter for the optimization technique among {0.1, 0.5, 1.0} to find optimal CRFs as we do in LSTMs.

Table 3 presents feature-set-level cross-validation results. LSTMs with the hyperparameter configuration of 64 hidden units and 0.7 dropout rate achieve the highest predictive accuracy (34.1%), and CRFs trained with the regularization parameters of 0.5 achieved the second highest accuracy (32.2%). We use raw correct and incorrect prediction counts to calculate accuracy rates rather than reporting fold-based averaged accuracy rates, in an effort to avoid the potential for skew brought on by the wide variation in the number of data points per student (min: 3; max: 41).

Table 3. Student-level leave-one-out cross validation results across feature sets (64 hidden units and 0.7 dropout rate for LSTMs and 0.5 regularization parameter for CRFs).

	LSTMs	CRFs
GSRs	28.0%	19.9%
AUs	21.8%	25.6%
GAMEs	29.4%	**32.2%**
GSRs / AUs	26.1%	22.3%
AUs / GAMEs	**34.1%**	30.8%
GSRs / GAMEs	29.9%	29.4%
GSRs / AUs / GAMEs	31.3%	27.0%

In the evaluation, LSTMs that achieve the highest predictive accuracy utilize AUs and GAMEs ($LSTM_{AU/GAME}$), the accuracy of which constitutes a 43.9% marginal improvement over the baseline accuracy (23.7%). Note that the baseline accuracy is different from Table 1, because it is influenced by the random split made in cross validation. We conducted a Wilcoxon signed rank, a non-parametric statistical test for two related samples, to compare cross-validation results between the $LSTM_{AU/GAME}$ and the majority class baseline per fold. The test finds a statistically significant difference between $LSTM_{AU/GAME}$ and the baseline (Z=-2.25, p=0.024). The differences between $LSTM_{AU/GAME}$ and the best performing CRFs (p=0.67) and between the CRFs and the baseline (p=0.095) are not statistically significant.

It is noteworthy that AUs by themselves do not achieve a high predictive accuracy. This can be partially explained by noting that the facial action unit data stream was often temporarily lost (a vector filled with zeros is used in this case for the missing data), usually when the subject's face was not properly situated within the camera screen. It is surprising, however, to see that partially-missing AUs synchronized with GAMEs data helped improve the prediction of the next virtual agent dialogue act by outperforming GAMEs models (Z=-1.71, p=0.088) as well as AUs models (Z=-2.24, p=0.025).

The $LSTM_{AU/GAME}$'s outperformance might be explained by the information available to the human wizards as they chose dialogue acts: they were able to watch the subject's game play as well as facial expressions during the interaction with the game, which together potentially influenced the dialogue decisions. On the other hand, the AUs likely characterize aspects of the subject's affective states, and they can contribute to the improved predictive performance synergistically with GAMEs in LSTMs.

Overall, GAMEs serve as a strong predictor relative to other independent data sources: GAMEs models (29.4%) outperform the other two independent models induced utilizing GSRs (28.0%) or AUs (21.8%); in the meantime, each feature set that leverages GAMEs in addition to other data sources outperforms the corresponding feature set without the GAMEs (e.g., GSRs, AUs, and GAMEs (31.3%) vs. GSRs and AUs (26.1%)). Sequences of actions in the GAMEs may reflect students' underlying cognitive states such as plans, goals, and knowledge during problem-solving activities [19, 20], which wizards attempted to address through their dialogue act choices. It is expected that LSTMs' capacity for hierarchical feature abstraction enables them to recognize these high-level patterns from low-level action sequences.

It is interesting to observe that GSRs by themselves outperform the baseline but incorporating GSRs with AUs and GAMEs (31.3%) does not outperform $LSTM_{AU/GAME}$ (34.1%). Although much of the previous research has used GSR data streams as evidence for modeling humans' affective and cognitive states [22], the findings of the study presented here suggest that GSR collected using wrist sensors may not be the most informative data source for predicting a human-operated virtual agent's next dialogue act, particularly when other data sources are available.

7. CONCLUSION AND FUTURE WORK

Dialogue modeling is a critical functionality for pedagogically adaptive virtual agents. This paper has presented two sequence-modeling approaches to classifying human wizards' dialogue moves when utilizing multimodal observation sequences. Both conditional random fields (CRFs) and long short-term memory networks (LSTMs) have demonstrated significant promise as effective modeling techniques on the sequential, parallel, multimodal data from game trace logs, galvanic skin response, and facial action units. Both CRFs and LSTMs outperform the majority class-based baseline with respect to predictive accuracy, while LSTMs achieve the highest predictive accuracy. Feature-level analyses of LSTMs suggest that even incomplete facial action unit data can augment LSTMs' predictive performance along with game trace logs, while game trace logs serve as strong predictor in both computational approaches. Along with achieving a substantial improvement in the use of sequence labeling techniques, this work suggests a number of directions for future work.

First, it will be important to extend the current models to determine the timing of dialogue acts. Together with the current work, this will further enhance the potential capacity for intelligent virtual agents to provide adaptive pedagogical support. Second, it will be important to examine the relationships between students' cognition and affect as perceived by human wizards, and to investigate how they influence wizards' dialogue decision-making. Because multimodal interaction data may reflect students' affective and cognitive states, identifying the relationship between student models and dialogue acts can guide the design of advanced tutorial dialogue management capabilities for pedagogical agents.

8. ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation through Grant CHS-1409639. Any opinions, findings, conclusions, or recommendations expressed are those of the participants, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

9. REFERENCES

[1] Baker, R., D'Mello, S., Rodrigo, M.M. and Graesser, A. 2010. Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. *International Journal of Human Computer Studies*. 68, 4, 223–241.

[2] Boyer, K., Phillips, R., Ha, E., Wallis, M., Vouk, M. and Lester, J. 2010. Leveraging Hidden Dialogue State to Select Tutorial Moves. *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*. 66–73.

[3] Chi, M., Vanlehn, K., Litman, D. and Jordan, P. 2011. An evaluation of pedagogical tutorial tactics for a natural language tutoring system: A reinforcement learning approach. *International Journal of Artificial Intelligence in Education*. 21, 1-2, 83–113.

[4] Chou, C.Y., Chan, T.W. and Lin, C.J. 2003. Redefining the learning companion: The past, present, and future of educational agents. *Computers and Education*. 40, 3, 255–269.

[5] D'Mello, S.K., Olney, A. and Person, N.K. 2010. Mining Collaborative Patterns in Tutorial Dialogues. *Journal of Educational Data Mining*. 2, 1, 1–37.

[6] Dawson, M.E., Schell, A.M. and Filion, D.L. 2007. The Electrodermal System. *The Handbook of Psychophysiology*. 200–223.

[7] DeVault, D. et al. 2014. SimSensei Kiosk : A Virtual Human Interviewer for Healthcare Decision Support. *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*. 1061–1068.

[8] Dweck, C.S. 2002. The development of ability conceptions.

[9] Ekman, P. and Friesen, W. V 1977. Facial action coding system.

[10] Forbes-Riley, K. and Litman, D. 2012. Adapting to Multiple Affective States in Spoken Dialogue. *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. 217–226.

[11] Graves, A. 2012. *Supervised sequence labelling with recurrent neural networks*. Springer.

[12] Hardy, M., Wiebe, E., Grafsgaard, J., Boyer, K. and Lester, J. 2013. Physiological Responses to Events During Training: Use of Skin Conductance to Inform Future Adaptive Learning Systems. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. 2101–2105.

[13] Johnson, W.L. 2010. Serious use of a serious game for language learning. *International Journal of Artificial Intelligence in Education*. 20, 175–195.

[14] Johnson, W.L. and Lester, J.C. 2015. Face-to-Face Interaction with Pedagogical Agents, Twenty Years Later. *International Journal of Artificial Intelligence in Education*. 25, 25–36.

[15] Lacoste-Julien, S., Jaggi, M., Schmidt, M. and Pletscher, P. 2013. Block-Coordinate Frank-Wolfe Optimization for Structural SVMs. *Proceedings of the 30th International Conference on Machine Learning*. 28, 9.

[16] LeCun, Y., Bengio, Y. and Hinton, G. 2015. Deep Learning. *Nature*. 521, 7553, 436–444.

[17] Littlewort, G., Whitehill, J., Wu, T., Fasel, I., Frank, M., Movellan, J. and Bartlett, M. 2011. The Computer Expression Recognition Toolbox (CERT). *Automatic Face Gesture Recognition and Workshops (FG 2011)*. 298–305.

[18] Min, W., Frankosky, M., Mott, B., Rowe, J., Wiebe, E., Boyer, K. and Lester, J. 2015. DeepStealth: Leveraging Deep Learning Models for Stealth Assessment in Game-Based Learning Environments. *Proceedings of the 17th International Conference on Artificial Intelligence in Education*, 277–286.

[19] Min, W., Ha, E.Y., Rowe, J., Mott, B. and Lester, J. 2014. Deep Learning-Based Goal Recognition in Open-Ended Digital Games. *Proceedings of the 10th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. 37–43.

[20] Min, W., Mott, B., Rowe, J., Liu, B. and Lester, J. 2016. Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks. *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. In Press.

[21] Mitchell, C., Boyer, K. and Lester, J. 2013. Evaluating State Representations for Reinforcement Learning of Turn-Taking Policies in Tutorial Dialogue. *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. 339–343.

[22] Poh, M.Z., Swenson, N.C. and Picard, R.W. 2010. A wearable sensor for unobtrusive, long-term assessment of electrodermal activity. *IEEE Transactions on Biomedical Engineering*. 57, 5, 1243–1252.

[23] Rowe, J., Shores, L., Mott, B. and Lester, J. 2011. Integrating learning, problem solving, and engagement in narrative-centered learning environments. *International Journal of Artificial Intelligence in Education*. 21, 1-2, 115–133.

[24] Shute, V.J., D'Mello, S., Baker, R., Cho, K., Bosch, N., Ocumpaugh, J., Ventura, M. and Almeda, V. 2015. Modeling how incoming knowledge, persistence, affective states, and in-game progress influence student learning from an educational game. *Computers & Education*. 86, 224–235.

[25] Shute, V.J. and Ventura, M. 2013. *Measuring and supporting learning in games: Stealth assessment*. Cambridge, MA: The MIT Press.

[26] Sutton, C. and McCallum, A. 2012. An Introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning*. 4, 4, 267–373.

[27] Vail, A., Grafsgaard, J., Wiggins, J., Lester, J. and Boyer, K. 2014. Predicting Learning and Engagement in Tutorial Dialogue: A Personality-Based Model. *Proceedings of the 16th ACM International Conference on Multimodal Interaction*. 255–262.

Exploring the Impact of Data-driven Tutoring Methods on Students' Demonstrative Knowledge in Logic Problem Solving

Behrooz Mostafavi
North Carolina State University
Raleigh, NC 27695
bzmostaf@ncsu.edu

Tiffany Barnes
North Carolina State University
Raleigh, NC 27695
tmbarnes@ncsu.edu

ABSTRACT

We have been incrementally adding data-driven methods into the Deep Thought logic tutor for the purpose of creating a fully data-driven intelligent tutoring system. Our previous research has shown that the addition of data-driven hints, worked examples, and problem assignment can improve student performance and retention in the tutor. In this study, we investigate how the addition of these methods affects students' demonstrative knowledge of logic proof solving using their post-tutor examination scores. We have used data collected from three test conditions with different combinations of our data-driven additions to determine which methods are most beneficial to students who demonstrate higher or lower knowledge of the subject matter. Our results show that students who are assigned problems based on profiling proficiency compared to prior exemplary students with similar problem-solving behavior show higher examination scores overall, and the use of proficiency profiling increases retention and reduces the amount of time taken in-tutor for lower performing students in particular. The results from this study also helps differentiate the behavior of higher and lower performing students in tutor, which can allow quicker interventions for lower proficiency students.

Keywords

Data-driven Methods, Proficiency Profiling, Tutoring Systems

1. INTRODUCTION

We have been incrementally adding data-driven methods for problem assignment[9, 10], hint generation[3], and worked examples[11] to the Deep Thought logic tutor to create a fully data-driven tutoring system. While we have observed improvements in student retention and tutor scores with each of these additions, we have not studied the difference in post-tutor examinations when these methods are combined in different test conditions. We seek to understand how the specific methods of problem assignment and combination of hints and worked examples may have impacted student performance on related questions on the course midterm exam.

In this paper we compare two classrooms of students using different test conditions of Deep Thought, with different combinations of problem assignment, hints and worked examples. Students' knowledge of logic were evaluated in two problems on a mid-term exam, and these scores were used to differentiate high and low proficiency students for our analysis. The results from our analysis show that high performing students benefit most from problem-solving opportunities, while low performing students benefit most from problem assignment based on proficiency profiling, comparing current students to prior exemplary students with similar behavior. We conclude that the use of proficiency profiling is the most effective method for increasing retention and reducing time spent in the Deep Thought tutor, and result in higher overall examination scores. The results from this study also help differentiate the behavior of higher and lower performing students in tutor, allowing for quicker interventions for lower proficiency students who need additional instructional support.

2. RELATED WORK

Koedinger et al.[6] summarized the general process of intelligent tutoring systems: the system selects an activity for the student, evaluates each student action, suggest a course of action (either via hints, worked examples, or another form of feedback), and finally updates the system's evaluation of the student's skills. An effective tutor should adapt instruction according to the student's current knowledge level [1]. However, in order to make instructional decisions, most ITSs either use fixed pedagogical policies providing little adaptability, or expert-authored pedagogical rules based on existing instructional practices [1, 14]. Intelligent tutoring systems with data-driven methods can be more adaptive by leveraging previous student data in order to complete one or more of these steps. Data-driven approaches to making effective pedagogical decisions – in particular selecting problems, when to apply worked examples, and the type of hint or feedback to provide – would mostly bypass the need for expert involvement in creating and improving the effectiveness of ITSs. In practice, incorporating student data has been shown to increase learning efficiency and predict student behavior. This, in particular is why we use data-driven knowledge tracing (DKT) of rule applications within

the Deep Thought logic tutor to facilitate profiling of students' proficiency.

In the remainder of this section, we describe the Deep Thought logic tutor and the data-driven additions implemented. We then describe the system and data used to evaluate the effectiveness of these data-driven methods in Deep Thought. After reporting the results of this evaluation, we discuss the implications for future design decisions in the tutor, and present our conclusions.

2.1 The Deep Thought Tutor

We have been examining the potential for data-driven methods to improve learning gains in a complex problem solving domain by incrementally augmenting the Deep Thought logic tutor. Deep Thought is a tutor for graphically constructing propositional logic proofs. Deep Thought presents proof problems consisting of logical premises and a conclusion to be derived using logical axioms. Deep Thought is divided into 6 levels of logic proof problems. In previous work with the Deep Thought logic tutor, we have been implementing data-driven methods for several of the intelligent tutor steps. We implemented a data-driven mastery learning system (DDML) to track student actions and assign appropriate problems based on the student's current level of proficiency [9]. The problem set was split into two tracks: a high proficiency track and a low proficiency track for Levels 2–6, with Level 1 containing a common set of problems for initial track assignment. We tracked student actions throughout their time in the tutor, and in particular their application of logical rules to construct logic proofs. Based on their correct or incorrect application of logical rules, the DDML updated a set of rule scores, one score for each logical rule. At the end of each level, the students' rule scores were weighted based on expert-determined priorities; rules deemed by experts to be of high importance to solving the problems in that level were weighted higher than rules that were not. These weighted scores were summed together, and compared to the average rule scores in the previous semester's data; based on this comparison, students were assigned to the higher or lower proficiency path. We tested Deep Thought with the DDML incorporated and found students completed, on average, 79% of all six levels in the tutor assignment. Student retention rate was 55%. This was an improvement over the non-DDML version of Deep Thought (61% tutor completion on average, and 31% retention rate).

We later incorporated a data-driven proficiency profiler (the DDPP) to replace the expert-determined priorities[10][8]. The DDPP is a system that calculates student proficiency at the end of each level in Deep Thought based on how a given student performs in comparison to exemplars who employed similar problem solving strategies, with rule scores weighted as determined through principal component analysis (PCA). Based on how similar exemplary students were assigned in subsequent levels, the DDPP can determine the best proficiency level for a new student. In contrast to the DDML system previously employed, this proficiency calculation and rule weighting is entirely data-driven, with no expert involvement.

We determined similar problem solving strategies among the exemplars by clustering the exemplars' rule scores based on hierarchical clustering. Expert weighting was replaced by PCA of the frequency of the rules used for each exemplar for each level, accounting for 95% variance of the results. For each rule, its PCA coefficient is the new weight for that rule score. When a new student uses the tutor, the student's rule scores are calculated throughout the level. At the end of each level, the DDPP examines each student's individual rule score and assigns it to a cluster for that rule. The DDPP then finds which clusters the scores for the most important rules fall into for that level (based on the same PCA based weighting), and then classifies that student into a *type* based on the set of clusters the student matches. Finally the system assigns the student to a proficiency track based on data from the matching type of exemplars, and how those exemplars were placed in the next level. The more exemplars we have of a given type, the stronger the prediction we can make for a new student. In the event that a new student doesn't match an existing type in the exemplar data, the student's proficiency is calculated using the average scores, as in the original DDML system.

Providing hints to students in the course of an intelligent tutor as a possible form of step-based feedback has the potential to increase learning gains. Razzaq, Leena, and Heffernan [12] found that learning gains increased for students given on-demand hints in comparison to students who were provided hints proactively. In Deep Thought, the hint system used is called Hint Factory. Hint Factory is an automatic data-driven hint generator that converts an interaction network graph of student trace behavior into a Markov decision process (MDP) to automatically select on-demand hints for students upon request, based on their individual performance on specific problems. The MDP is data-driven, using actions logs from previous Deep Thought use in the classroom to assign weight to proof-state actions based on whether or not that action ultimately led to successful completion of the proof. These hints help students solve problems by suggesting what step should be taken next on a multi-step problem. Hint Factory has been implemented in the Deep Thought logic tutor to automatically deliver context-specific hints to students during problem-solving [4]. In a previous study Hint Factory was shown to provide context-specific hints over 80% of the time [3]. In a pilot study, Barnes & Stamper found that Hint Factory can provide sufficient, correct, and appropriate hints for the Deep Thought Logic tutor and help students to solve more logic proof problems in the same span of time [4]. However, we currently cannot determine the effect hints would have in addition to the DDML or DDPP; so far, students using either of those versions of Deep Thought did not use hints often enough for any meaningful analysis.

Adding worked examples as a supplement to traditional problem solving can also be beneficial [2, 13]. Hilbert and Renkl [5] found that improved learning outcomes occurred when providing worked examples with a prompt, and proposed that this was due to allowing the students to have a greater cognitive load at once. McLaren and Isotani [7] compared three tutors using all worked examples, all traditional unguided problem solving, and a mix of worked examples and problem solving. Each group achieved similar learning gains, but the students who were given all worked examples required less time to achieve those gains. We added worked

examples to the version of Deep Thought with the DDML incorporated[11]. Worked examples were generated based on previous best student solutions, and procedurally annotated. They were presented to students randomly on a per-problem basis, based on the number of problems they had solved in that level already. We found that student retention overall was 90%, and students completed 94% of the tutor on average. This percentage was significantly higher than that of the DDML alone.

3. METHODS

Deep Thought was used as a mandatory homework assignment by students in an undergraduate "discrete mathematics for computer scientists" course in Fall 2015 and Spring 2016. Students in the two semesters were taught by different instructors. Students were assigned Levels 1–6 of Deep Thought for full credit, with partial credit awarded proportional to the number of levels completed. For this study, we compare the data from three Deep Thought test conditions used across the two semesters to differentiate the effect of our data-driven methods on student performance.

The first group evaluated for this study were assigned only problem-solving opportunities (PS group, $n = 26$). The problem assignment system used was the DDML system described in the previous section, where students were assessed between levels and placed on either a high or low proficiency track in the next level. This group of students were taken only from the Fall 2015 semester, as there existed no equivalent test condition in Spring 2016.

The second group of students were randomly assigned either problem-solving opportunities or worked examples of the same problems within each level (PS/WE group, $n = 179$), with the number of problem-solving opportunities controlled to match the number of problems solved by the PS group. Like the PS group, the PS/WE group were assigned proficiency tracks using the DDML. However, because individual rule application scores were updated at each step in worked examples as if a student had applied that rule in while problem solving, most students were consistently assigned to the high track in most levels, and were only assigned the low track when their individual performance was below satisfactory. This group of students were taken from both the Fall 2015 and Spring 2016 semesters.

The third group of students were randomly assigned problem-solving opportunities or worked examples in the same manner as the PS/WE group, but with the DDPP method assigning proficiency tracks instead of the DDML, where students were assigned the same proficiency track as prior students who most closely matched their rule application behavior (DDPP group, $n = 61$). This group of students were also taken from both the Fall 2015 and Spring 2016 semesters. Students in all three groups had access to on-demand hints.

All students were evaluated using two proof problem questions as part of a mid-term examination, which was used as a post-test for this study. Students performance in the post-test for both Fall 2015 and Spring 2016 were graded by the same teaching assistant, ensuring consistent evaluation across all results. Students were separated for evaluation by performance on the post-test and by the predominant track level in Deep Thought. The post-test was a set of two proofs students had to solve on paper for a midterm exam. These questions were hand-graded with partial credit given based on the percentage of the proofs completed and points taken off for misapplication of rules and skipping non-trivial rules. We considered two performance levels: post-test scores greater than or equal to 80% (AB), or less than 80% (CDF). The post-test scores mark the final evaluation of students' ability to solve proof problems, and occurs immediately following the Deep Thought tutor homework assignment.

The second dimension we studied was the proportion of high to low proficiency track levels the students completed. Students who were assigned to the high proficiency track in a level had the ability to finish on either the high or low proficiency track depending on the number of problems skipped within that level. Students who completed more levels on the high track than the low track were marked as high track students, and students who completed more levels on the low track than the high track were marked as low track students. The track assignments indicate the number and complexity of problems students received, with the low track having more problems of lower complexity, and the high track having fewer problems of higher complexity. The tracks were designed so that students would have a similar number of rule applications across the tracks, even though the number of problems differs. Typically, the low track has three problems with expert solutions using 5 rule applications, and the high track has 2 problems with expert solutions using 7 – 8 rule applications - meaning that both tracks minimally required about 15 total rule applications (though students typically used more).

In addition to post-test and predominant track level, we examined total time in tutor, average time spent per problem, percentage of correct rule applications out of all rule applications, and the total number of rule applications. We also looked at ancillary behaviors (hint usage, skipped problems, and reference requests) that could differentiate high and low performing students. We compared these metrics to better understand the impact of worked examples, hints, and data-driven track selection on student performance. The results of this descriptive analysis are presented in the next section.

4. RESULTS

Table 1 displays the percentage of AB students in each of the PS, PS/WE, and DDPP groups for all students, as well as students who completed the majority of the tutor in either the high or low tracks. Table 1 also displays the percentage of students in each group and each track who dropped out of the tutor before full completion, as this is one of the metrics we have used to judge the effectiveness of our data-driven methods. In our previous work using the same version of Deep Thought, we found that students completed 94% of the tutor on average, with a retention rate of 90%. The average percent tutor completion for the groups in this study were consistent with these numbers (PS: 95%, PS/WE: 93%, DDPP: 94%).

The first interesting result of note is that the percentage of students who performed better on the post-test was higher

Table 1: Percentage of AB Students and Percentage of dropped students in the PS, PS/WE, and DDPP groups.

Condition		ALL	High Track	Low Track
	n	*% AB Students*		
PS	*26*	65.38	71.43	63.16
PS/WE	*179*	49.72	52.35	36.67
DDPP	*61*	63.93	66.67	61.76
	n	*% Dropped Students*		
PS	*26*	3.85	0.00	5.26
PS/WE	*179*	11.73	6.71	36.67
DDPP	*61*	9.84	11.11	8.82

for for the PS (65%) and DDPP (64%) groups than for the PS/WE group (50%), across all the students, as well as within the high and low track groups. In the PS group, students who completed more levels on the high track displayed a higher overall proficiency of the subject matter than those who finished more often on the low track (71% vs 63%, respectively), as did students in the PS/WE group (52% vs 37%).

However, students in the DDPP group showed a consistent level of proficiency regardless of the tracks completed (66% vs 61%), which makes sense considering that these students were matched to previous successful students who displayed similar rule-application behavior, and had a more even placement within the high and low tracks compared to the PS group, who had even placement among tracks, but within the context of their own performance compared to expert-decided thresholds. The DDPP group also had higher placement compared to the PS/WE group, who were placed on the high track much more often than not due to the inclusion of worked examples. A Kruskal-Wallis test for one-way analysis of variance showed no significant difference between groups ($p = 0.22$).

Students also had a higher retention rate in both the PS (4%) and DDPP (10%) groups compared to the PS/WE group (12%). It is especially interesting to see the drop rate among low track students in the PS/WE group, who had a much lower retention rate among all the students in the study. Because students in the PS/WE group were more often that not placed in the high track in each level, for students to end up on the low track indicates a high level of problem-skipping among these students. We can conclude that low performing students who are not intelligently assigned problems based on their problem-solving performance appear to gain little from worked examples.

While it may be tempting to declare problem-solving opportunities with no worked examples as the best performing pedagogical choice among the three groups based on these numbers alone, a look into additional performance metrics gives some more insight. Table 2 presents the amount of time spent in tutor and on each problem, as well as the percentage of correct and total rule applications for each group, separated by track. The numbers presented are the median values for each metric, since the distributions of scores were highly skewed and non-normal, and none of the differences were significant due to low sample size within each subgroup.

As shown in Table 2, among AB students in all three groups, the total time spent in tutor appears similar, although the mean time for high-track students was lower for DDPP (M = 3.95hr, SD = 6.21hr) compared to PS/WE ($M = 4.46hr$, $SD = 9.13hr$) and PS ($M = 6.66hr$, $SD = 9.91hr$). The mean time for low-track students was lower for PS ($M = 4.63hr$, $SD = 9.55hr$) and DDPP ($M = 5.48hr$, $SD = 5.42hr$) than the PS/WE ($M = 7.74$, $SD = 9.76$). The means of average problem time, percentage of correct rule applications, and number of rule applications were consistent with the median values presented in Table 2 across all three groups. Note that low-track students in the PS/WE groups had the lowest percentage of correct rule applications, and the highest number of total rule applications among all the groups. This means they are doing more work, but a lower percentage of it is correct.

As shown in Table 2, among CDF students in all three groups, the total time spent in tutor is dramatically different, with PS spending 3 to 4 times as long in the tutor than PS/WE and DDPP groups. This ratio is also similar in the average problem time for high and low track students, and the number of total rule applications for high track students. Therefore, while problem-solving only (PS) may have a slightly higher overall success rate in helping students learn proof problem solving and remain in the tutor than the DDPP students, for students who are less prepared, PS results in a much higher time spent in the tutor, with little return on the time investment. Therefore, for students who have a better grasp of the subject matter, pure problem-solving may offer a slightly better option for getting through the assigned tutor, although the differences between problem solving, problem solving and worked examples, and proficiency profiled assigned problem solving and worked examples are minimal. However, for less prepared students, pure problem-solving opportunities offer little to guide students to higher understanding of the material, and in general, the DDPP offers a much better path to completing the tutor in far less time for both AB and CDF students, giving students the opportunity to encounter all the subject matter and have a greater chance of learning the material, resulting in higher overall post-test scores.

Completing the tutor assignment is important for students; however, since we want to make sure that students are learning the material well, mid-term examination scores are ultimately a higher gauge for learning success. Among all the experimental groups in this study, at most 65% of students were performing at A or B grade level on the mid-term examination. We would like to increase this percentage of AB students, so the question at this point is: Is it possible for us to predict low exam scores based on in-tutor data for early intervention?

We first look at the differences between AB and CDF students in Table 2, with the assumption that the DDPP method offers the best overall chance of success for students. For high track students, total tutor time, average problem time, percentage of correct rule applications, and total rule applications are consistent between AB and CDF students. However, for low track students, average problem time, percentage of correct rule applications, and total rule applications show a higher difference. CDF students spent twice as

Table 2: Total Time, Average Problem Time, Percentage of Correct Rule Applications, and Total Rule Applications for AB and CDF students in the PS, PS/WE, and DDPP groups, separated by High and Low Track. The numbers listed are all median values.

		AB STUDENTS				CDF STUDENTS		
		PS	**PS/WE**	**DDPP**		**PS**	**PS/WE**	**DDPP**
HIGH TRACK	***n***	*5*	*78*	*18*	***n***	*2*	*71*	*9*
Total Tutor Time (hr)		2.47	2.37	2.80		12.8	3.75	3.17
Average Problem Time (min)		9.89	11.1	12.1		52.3	18.4	16.0
% Correct Rule Applications		60.8	63.5	58.5		64.1	56.9	62.3
Total Rule Applications		258	214	203		471	255	204
LOW TRACK	***n***	*12*	*11*	*21*	***n***	*7*	*19*	*13*
Total Tutor Time (hr)		1.80	3.33	3.67		17.2	5.96	4.98
Average Problem Time (min)		6.76	15.2	15.0		60.1	25.0	30.4
% Correct Rule Applications		68.8	45.5	57.0		48.7	45.7	47.0
Total Rule Applications		201	404	291		382	394	389

long on average per problem than AB students, and applied rules correctly less than half of the time, while AB students applied rules more than half of the time. CDF students also attempted applying rules 25% more overall than AB students.

Since the performance differences between AB and CDF students are not as apparent for high track students, we look at ancillary tutor behavior to make a better distinction. Table 3 shows the number of requested hints, the number of skipped problems, and the number of rule reference requests (descriptions of logic rule operations) made by students in all groups. For the DDPP group, the most apparent difference among AB and CDF students are the number of hints requested, with the CDF group requesting 32 hints ($M = 50$, $SD = 57$) compared to 17 ($M = 32$, $SD = 42$) for the AB group. This difference in hints requested between AB and CDF students is also consistent across all groups and both high and low track students. We conclude that for high track students, we can differentiate between higher and lower proficiency students using hint request behavior, and for low track students, we can differentiate higher and lower proficiency students using the amount of time spent on average per problem and the percentage of correct rule applications. This allows the possibility of making an intervention during a student's progress through Deep Thought in the case that a student requires additional feedback or aid from an instructor due to a lesser understanding of the subject matter.

Table 3: Number of Hints, number of Skips, and number of Rule Reference requests for AB and CDF students in the PS, PS/WE, and DDPP groups, separated by High and Low Track. The numbers listed are all median values.

	PS		**PS/WE**		**DDPP**	
HIGH	**AB**	**CDF**	**AB**	**CDF**	**AB**	**CDF**
# Hint	95	166	12	26	17	32
# Skip	5	16	1	1	0	2
# Ref	151	168	76	145	111	92
LOW	**AB**	**CDF**	**AB**	**CDF**	**AB**	**CDF**
# Hint	30	104	31	44	19	26
#Skip	1	0	30	24	3	15
# Ref	77	224	60	271	55	109

5. CONCLUSION

In this paper we compared two classrooms of students using different test conditions of Deep Thought, with different combinations of problem assignment (DDML or DDPP) and the addition of worked examples, for the purpose of understanding how the specific methods of problem assignment and combination of hints and worked examples affect high and low performing students, as evaluated using mid-term examination scores. We found that for higher proficiency students who have a firmer grasp of the subject matter, problem-solving opportunities offer the best chance of completing the tutor in a timely manner; however, the addition of worked examples does not significantly detract from these students' learning experience. The method of problem assignment (DDML or DDPP) does not have a noteworthy effect on high student performance.

For lower proficiency students, we found that problem-solving opportunities alone with DDML problem assignment offered little to guide students to higher understanding of the material, and greatly extended the amount of time students spent in the tutor with little learning benefit. The addition of worked examples helped these students get through the tutor faster, however these students had a lower retention rate than any other students and lower examination scores. We conclude from these results that updating our data-driven skill estimates equally for viewing or applying rules resulted in students being assigned to the high-track when they were not prepared to solve harder problems. With proficiency profiling – matching students to previously successful students and the paths they take through the tutor – we can reduce the amount of time spent in tutor, increase retention, and make better use of worked examples by giving them alongside problems that better match an individual student's proficiency level. This results in similar performance to problem solving alone in terms of retention and knowledge gained, but with a lot less time spent in the tutor for lower-proficiency students. We conclude that our DDPP method offers the best overall possibility of success for students completing the Deep Thought tutor in a timely manner, learning the subject matter, and performing well on post-tutor examinations.

6. ACKNOWLEDGEMENTS

This material is based on work supported by the National Science Foundation under Grants 1432156 and 0845997.

7. REFERENCES

[1] J. R. Anderson, A. T. Corbett, K. R. Koedinger, and R. Pelletier. Cognitive Tutors: Lessons Learned. *The journal of the learning sciences*, 4(2):167–207, 1995.

[2] R. K. Atkinson, S. J. Derry, A. Renkl, and D. Wortham. Learning from examples: Instructional principles from the worked examples research. *Review of educational research*, 70(2):181–214, 2000.

[3] T. Barnes, J. Stamper, L. Lehmann, and M. J. Croy. A pilot study on logic proof tutoring using hints generated from historical student data. In *Proceedings of the 1st International Conference on Educational Data Mining (EDM 2008)*, pages 197 – 201, 2008.

[4] T. Barnes, J. Stamper, L. Lehmann, and M. J. Croy. A pilot study on logic proof tutoring using hints generated from historical student data. In *Proceedings of the 1st International Conference on Educational Data Mining (EDM 2008)*, pages 197 – 201, 2008.

[5] T. S. Hilbert and A. Renkl. Learning how to use a computer-based concept-mapping tool: Self-explaining examples helps. *Computers in Human Behavior*, 25(2):267–274, 2009.

[6] K. R. Koedinger. New Potentials for Data-Driven Intelligent Tutoring System Development and Optimization. *AI Magazine*, 34(3):27–41, 2013.

[7] B. M. McLaren and S. Isotani. When is it best to learn with all worked examples? In *Artificial Intelligence in Education*, pages 222–229, 2011.

[8] B. Mostafavi and T. Barnes. Data-driven Proficiency Profiling - Proof of Concept. In *Proceedings of the 6th International Conference on Learning Analytics and Knowledge (LAK 2016). In Press.*, 2016.

[9] B. Mostafavi, M. Eagle, and T. Barnes. Towards data-driven mastery learning. In *Proceedings of the 5th International Conference on Learning Analytics and Knowledge (LAK 2015)*, pages 270–274, 2015.

[10] B. Mostafavi, Z. Liu, and T. Barnes. Data-driven Proficiency Profiling. In *Proceedings of the 8th International Conference on Educational Data Mining (EDM 2015)*, pages 335–341, 2015.

[11] B. Mostafavi, G. Zhou, C. Lynch, M. Chi, and T. Barnes. Data-driven Worked Examples Improve Retention and Completion in a Logic Tutor. In *Proceedings of the 17th International Conference on Artificial Intelligence in Education (AIED 2015)*, pages 726–729, 2015.

[12] L. Razzaq and N. T. Heffernan. Hints: is it better to give or wait to be asked? In *Proceedings of the 10th International Conference on Intelligent Tutoring Systems (ITS2010) Part 1*, pages 349–358. Springer, 2010.

[13] J. Sweller and G. A. Cooper. The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction*, 2(1):59–89, 1985.

[14] K. VanLehn. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education*, 16(2):227–265, 2006.

Properties and Applications of Wrong Answers in Online Educational Systems

Radek Pelánek
Masaryk University Brno
xpelanek@mail.muni.cz

Jiří Řihák
Masaryk University Brno
thran@mail.muni.cz

ABSTRACT

In online educational systems we can easily collect and analyze extensive data about student learning. Current practice, however, focuses only on some aspects of these data, particularly on correctness of students answers. When a student answers incorrectly, the submitted wrong answer can give us valuable information. We provide an overview of possible applications of wrong answers and analyze wrong answers from three different educational systems (geography, anatomy, basic arithmetic). Using this cross-system comparison we illustrate some common properties of wrong answers. We also propose techniques for processing of wrong answers and their visualization, particularly an approach to item clustering based on community detection in a confusion graph.

1. INTRODUCTION

A key advantage of computerized educational systems is their potential for personalization. By analyzing students' answers we can estimate their knowledge using student modeling techniques and adapt the behaviour of a system to the needs of individual students. Student models [6] typically utilize only information about correctness of answers. Online systems, however, collect (or can easily collect) much richer information, e.g., timing information [18] and specific details about answers and individual steps. In this work we focus on analysis of wrong answers.

Wrong or incomplete answers from online educational systems have been studied previously, but mostly just as a supplementary analysis to other research interests. For example, analysis of programming assignments in MOOCs [9, 14] shows that the distribution of wrong answers is highly skewed, containing few very common wrong answers. This research does not, however, focus on analysis of wrong answers, but rather on finding similar or equivalent solutions and their visualizations (as there are many ways how to write the same program) [7].

The observation that distribution of wrong answers is highly skewed holds not only for programming assignments, but also for other domains. For example, common wrong answers have been used for student modeling in mathematics [29], but this work uses only information about whether the wrong answer is common or not, it does not utilize actual values of wrong answers. Specific student answers were also modeled [8], but authors present only overall accuracy of the proposed model without discussion of specific mistakes.

Data analysis techniques has been used for analysis of mathematical errors with the goal of classification (explanation) of answers [13, 24]. The results show that it is possible to classify most wrong answers into one of few categories. Other data-driven techniques in educational data mining have focused mainly on programming assignments [10, 21]. Rather than "wrong answers" they utilize "incomplete solutions" and use them for automatic generation of hints (changes towards a correct solution).

In the wider context, wrong answers are related to misconceptions, which are intensively studied in pedagogical literature, e.g., misconceptions in mathematics [26] or chemistry [22]. This line of research focuses on understanding "buggy rules" used by students [4]. These rules are useful not just for educating teachers about student thinking, but also in development of intelligent tutoring systems.They can be also used as a basis of erroneous examples [1, 11]. Research in this direction is typically based on expert insight using only relatively small (and often qualitative) data and the focus is typically on complex skills.

In this work we focus on automatic techniques for analysis of large quantitative data, dealing with simple skills (learning of declarative knowledge and simple procedures). We describe analysis of wrong answers from three educational systems. Although the used systems share similar basic principles they cover widely different domains (geography, anatomy, basic arithmetic) and different learner populations (from kindergarten to university students). Thanks to the size of the used data set (millions of answers), results provide interesting insights into properties and potential of wrong answers. We describe specific examples of analysis and propose novel techniques for analysis and visualization of wrong answers. A key observation is that wrong answers in our three domain (geography, anatomy, basic arithmetic) share many properties and thus it should be feasible to carry insights and analysis techniques across domains.

2. POTENTIAL APPLICATIONS OF WRONG ANSWERS

In this section we outline potential applications of wrong answers. The presented applications are rather general and for a specific application they need to be more precisely quantified. In the next section we provide such specific analysis for three particular domains.

2.1 Student and Domain Modeling

Student and skill models [6] typically utilize only binary information about correctness of an answer (correct/incorrect). A more thorough analysis of wrong answers may improve student and skill modeling in several directions.

In modeling of cognitive skills, wrong answers may help to distinguish between absence of understanding and slips (careless errors, typos). Highly uncommon wrong answer is more likely to be a careless error, whereas common wrong answer is more likely to be a genuine mistake (unless caused by poorly designed user interface). Wrong answers may also be indicative of the level of knowledge and strategies that students are using. Consider for example a multiplication 5×5: a student A answers quickly 30, whereas a student B answers 24 after a long time. This may indicate that the student A retrieved the answer (incorrectly) from declarative memory, whereas the student B made an error in a procedural strategy. Wrong answers can thus be useful for modeling cognitive processes of learners [27]. Moreover, they may be useful also for modeling affect and motivation [29]. Irrelevant, highly uncommon wrong answers (particularly when repeated and quickly delivered) are probably indication of disengagement rather than lack of knowledge.

Wrong answer may be useful also for domain modeling. Common wrong answers may indicate relations between topics and thus may be used for automatic detection of knowledge components. Even through these may be misconceived relations, when they are common, they may be useful for student modeling. Relations between items based on wrong answers may also be taken into account in the design of the user interface or in the item selection algorithm. Wrong answers can also be used for student clustering – different groups of students make different types of mistakes and need different treatment from the educational system (e.g., students with dyslexia or dyscalculia).

2.2 Construction of Items and Hints

A basic observation about wrong answers, which seems to be valid in many different domains, is that the distribution of wrong answers is often highly skewed, i.e., some mistakes are much more common then others. This feature of wrong answers is potentially very useful for construction of questions and hints (both manual and automatic).

Common wrong answers may highlight student misconceptions and thus provide inspiration for new items (problems). In the case of items with simple structure, wrong answers may even be used automatically, e.g., as competitive distractors in multiple choice questions [16]. Previous work [1, 11] explored the possibility of using erroneous examples in education. Common wrong answers provide useful material for creation of such examples.

Wrong answers may also be useful for development of hints, feedback to students, and other scaffolding aids. If the hints are developed manually by experts, wrong answers provide good way to prioritize the expensive work of an expert. Due to the skewed distribution of wrong answers it may be possible to quickly provide answer-specific feedback to most answers even in open environments [9]. It is also possible to generate hints automatically based on actions of other students with the same wrong answer [23].

2.3 Feedback for Learners, Teachers, and Tool Developers

Analysis of wrong answers can also bring more pragmatic advantages. A useful feature of personalized educational systems is an overview of mistakes made by a learner or a class. Such an overview can serve for example as a base for a review session. Teachers may use such overview to quickly detect common problems of their students and thus focus on problematic parts in classroom time or in personal consultations.

For tool developers common wrong answers may be useful as an indicator of problems with a user interface. For example, in a prototype of one of the systems used in this study there was a common wrong answer "1" in cases where the answer should have been "10". This turned out to be a user interface issue – the system was expecting a single click on a "10" button, whereas users were trying to click buttons "1" and "0".

For these types of applications, basic analysis of wrong answers should be easily accessible in educational systems for both teachers and system developers. Since there can be a large number of mistakes, it is important to make the listing of mistakes easy to navigate. To achieve this goal, we need to understand common features of wrong answers.

3. ANALYSIS OF WRONG ANSWERS

After the general discussion of properties and possible applications of wrong answers, we turn to analysis of specific data.

3.1 Used Systems and Data

The used systems cover three different domains (geography, anatomy, basic arithmetic) and are used by very different learners, but they have been developed by the same research group and share the basic principles. All of them focus on adaptive practice of declarative knowledge or simple procedures. Systems estimate learners' knowledge and based on these estimates they adaptively select questions of suitable difficulty. They use a target success rate (e.g., 75%) and adaptively selects questions in such a way that the learners' achieved performance is close to this target.

The used questions are either multiple-choice questions or "open questions" – either a free text answer or selection of any item from a provided context (e.g., "select Rwanda on the map of all African states"). For the analysis we use only answers to open questions, since the used multiple-choice questions have adaptively chosen distractors and this feature makes analysis difficult (due to the presence of feedback loops [19]).

The first system is Outline Maps (`outlinemaps.org`) for practice of geography facts (e.g., names and locations of countries, cities, mountains). Details of the behaviour of the system are described in [15, 16]. The used data set contains more than 10 million answers (with more than 1 million wrong answers) and is publicly available [17]. This data set is the largest of the three used data sets and it is at the core of the presented analysis. The application is currently used by hundreds of learners per day, majority of learners is from the Czech Republic since the interface was originally only in Czech. The geographical origin and language of students clearly influence interpretation of below presented results. However, our main point is not interpretation of particular results, but rather illustration of different insight that can be gained by the analysis of the data.

The second system is Practice Anatomy for practicing human anatomy (`practiceanatomy.com`). The main target audience of the system consists of junior medical students preparing for their anatomy exams. Currently, the system offers practice of more than 1800 items organized into 14 organ systems and 9 body parts. Learners can practice a selected organ system or a body part, or specify a more advanced practice filter as an intersection of a set of organ systems and a set of body parts. The system is available in Czech (with Latin terminology) and English. Most users are from the Czech republic. The analyzed data set contains over 380 000 answers.

The third system is MatMat (`matmat.cz`) for practice of basic arithmetic; its functionality is similar to for example Math Garden [24]. The system contains examples divided into 5 high level concepts (counting, addition, subtraction, multiplication, division), each of these concepts contains around 50-700 items, over 2 000 items in total. The system behaviour and the used student modeling approach are described in [28]. The analyzed data set contains over 180 000 answers.

Student knowledge and mistakes in the used domains have been analyzed before, e.g., recall and mistakes in knowledge of US states [20] or knowledge of Europe by Turkish students [25]. These works focused on difficulty of recall of individual countries and on factors which influence this difficulty (e.g., borders), they did not analyze wrong answers. Moreover, we use a data set that is orders of magnitudes larger than those used in previous research on geography knowledge. The domain of basic arithmetic has been studied intensively before, even with the focus on mistakes. A well-known example is the repair theory [4] with case study for subtraction problems. Particularly multiplication has been studied in detail, e.g., description of effects influencing difficulty (size effect, five effect, tie effect), connectionist model of retrieval [27], classification or errors [5, 24]. Our contribution in this domain is mainly in aligning the results with analysis from different domains (learning declarative knowledge in geography and anatomy).

3.2 Common Wrong Answers

Generally the distribution of wrong answers is highly skewed, most wrong answers are comprised from just few items. Analysis of commonly confused countries shows that the most important factors are whether the countries have common border, if they have similar size (important factor particularly if they have a common border) and whether their name starts with the same first letter (important factor particularly if they do not have a common border). There are differences between the skewness of the distribution of wrong answers for individual items. For some countries there are few very typical mistakes – for Bulgaria more than 40% of wrong answers are Romania, for Finland the two most common wrong answers (Sweden and Norway) comprise nearly three quarters of wrong answers. Some countries, however, have much more even distribution of wrong answers, e.g., for Switzerland or Croatia the most common mistake comprises only 10% of wrong answers.

The context of questions is also important. In the used system countries can be practiced either in the context of a single continent or of the whole world. In most cases the mistakes on the world map are within the same continent (i.e., the wrong answers on the world map are very similar to wrong answers within the continent map). There is, however, nontrivial number of exceptions, for example: countries with similar names, e.g., Guinea, Guyana, and Papua New Guinea, which have confusingly similar names and are on three different continents; countries close to continent borders, e.g., Turkey is confused with European countries and Arab countries in Africa and Asia confused; islands are confused together, e.g., Madagascar is not confused with other African countries, but with other islands. These examples illustrate the importance of proper practice context for some items, e.g., it is not very useful to practice Madagascar on the map of Africa, Madagascar should be practiced mainly on the map of the whole world. Such results can have direct consequences for the design of the behaviour of educational systems.

The data from the MatMat application contain similar patterns – the distribution of wrong answers is skewed, but the skewness of the distribution differs among items. Some items have very typical wrong answer (e.g., $1 \times 1 = 2, 4 \times 9 = 32$), for other items wrong answers are more uniformly distributed (e.g., 6×8 with answers 42, 54, 56, 64, 78). Previous work [24] has analyzed classification of errors in basic arithmetic (particularly in multiplication), using categories like near miss (± 1), typo, operation error, or operand related error. In agreement with previous research [13, 24], large part of wrong answers fit into one of these categories, and the dominant categories are as expected – for counting and addition the dominant error type is "near miss", whereas for multiplication a common error is operand related, e.g., $4 \times 9 = 32$ (which is 4×8). There are, however, interesting differences between items of the same type. For division the typical mistake is "near miss" (± 1). For division by 1 and 10, however, the typical mistakes are rather answers 1 and 10; for items of the type N/N common wrong answers are N or 0. For small operands (e.g., 4/2) operation errors (multiplication instead of division) sometimes occur, whereas this does not happen for larger operands (e.g., 54/6).

3.3 Categories of Wrong Answers

To provide a more quantitative analysis and comparison across educational systems, we define a coarse classification of wrong answers and analyze properties of individual categories. We propose the following classification of wrong an-

swers into four categories (note that the defined categories can be seen as "degrees of wrongness" of an answer with a natural ordering). *TopWA* is the most common wrong answer for a given item. *CWA* is a common wrong answer other then the most common one (as a definition of "common" we require that the number of occurrences is more then 5% of all wrong answers for the given item, it must also be larger than 1). *Other* is any nonempty answer that is not common. *Missing* is an empty answer. Previous research [29] used 10% bound for definition of common wrong answers, but they did not treat the top wrong answer separately.

Figure 1 (top) shows distribution of answers among these classes. Although there are some differences between the used systems (respectively specific maps in the geography system), overall the distribution is quite balanced, i.e., the used definitions of classes provide reasonable partition of wrong answers. The rest of Figure 1 shows characteristics of student behaviour related to answers from individual categories. Since in this work we are interested mainly in relative comparison among types of answers (and not among systems), the results are normalized with respect to correct answers (for each system). The reported characteristics are computed globally. We have also analyzed more detailed results (e.g., for specific practice contexts like European countries or one digit multiplication), the results show similar trends.

The results show clear trends that are very similar across the three used systems. The median response time is larger for more wrong answers, with the exception of missing answers. The probability of leaving the system directly after an answer is much higher for wrong answers than for correct answers. Also within the wrong answers there is a clear trend (the probability of leaving increasing with wrongness). Finally, the last two graphs analyze future success of a student; the probability of success on the next question about the same item, the probability of success on the next question within the system (global). In both cases there the probability of future success decreases with wrongness of the current answer.

We see that there are systematic differences between different types of wrong answers. The general nature of these differences is rather intuitive, the main interesting aspects of these results are the similarity of results across three different domains and the consistently linear nature of these relationships, i.e., we can say that the distance between *TopWA* and *CWA* is the same as the distance between *CWA* and *Other*. The bottom line is that the wrongness of answers can be treated as an interval variable and it may be useful to utilize it as such for student modeling (for modeling both knowledge and affect).

3.4 Confusion Graph and Item Clustering

So far we have analyzed wrong answers for each item separately. But mistakes for individual items are clearly interconnected. We can analyze these interconnections with a "confusion graph" (a similar analysis has been done previously for the domain of statistics [12], but for much smaller data). In a confusion graph nodes are individual items, and edges correspond to wrong answers – we consider a weighted graph where a weight of an edge (u, v) is given by a frequency of a particular wrong answer v among all wrong answers on an item u. This definition leads to a directed graph, to obtain an undirected graph we compute the weight of an undirected edge by averaging the weights of the corresponding directed edges.

Figure 2 shows the confusion graph for European countries. The confusion graph contains distinct clusters of items, this observation holds also for confusion graphs of other practice contexts in the used systems. To automatically detect these clusters we use a community detection algorithm [3]. The resulting clusters are meaningful and can provide useful insight for teachers and developers of educational system (Figure 2 for an illustration). The presented clustering was obtained by off-the-shelf implementation of the community detection algorithm [2] without any tuning. For a specific application of such clustering it may be useful to experiment with different community detection algorithms and specific definitions of the confusion graph.

3.5 Other Properties of Wrong Answers

Wrong answer may help us to (quickly) differentiate between different groups of users. For example in the geography domain we can see some important differences in wrong answers of students of different geographical origin, e.g., confusions between Slovakia and Slovenia, which is much more common mistake for US students than for Czech students, or wrong answers for Belarus (Bulgaria for US students, Ukraine for Czech students).

Wrong answers differ in their "persistence", i.e., probability that the mistake will be repeated (by the same student) in future. For example, consider wrong answers for Ireland. United Kingdom is more probable mistake than Italy, but the second one is more likely to persists. Other similar examples are Moldova (answers Macedonia versus Kosovo) or Benin (answers Burundi versus Ghana). Some mistakes are very likely to be repeated, e.g., confusion between Zambia and Zimbabwe, Gambia and Senegal, or Guinea-Bissau and Burkina Faso.

4. CONCLUSIONS

Our analysis suggests that wrong answers are underused resource in online educational systems. They are easy to collect and can provide interesting insight applicable in many different ways (student modeling, automatic question and hint construction, feedback and inspiration for teachers and system developers). We provide a systematic overview of potential applications of wrong answers and many illustrative examples of interesting insights from educational applications.

We also propose specific novel approaches to analysis and utilization of wrong answers, particularly a classification of wrong answers into four categories (which can be treated as "degrees of wrongness") and clustering of items using a confusion graph (based on wrong answers) and a community detection algorithm. The results of analysis from three different domains (geography, anatomy, basic arithmetic) show that properties of wrong answers are rather consistent and thus the developed approaches should be applicable also for other domains.

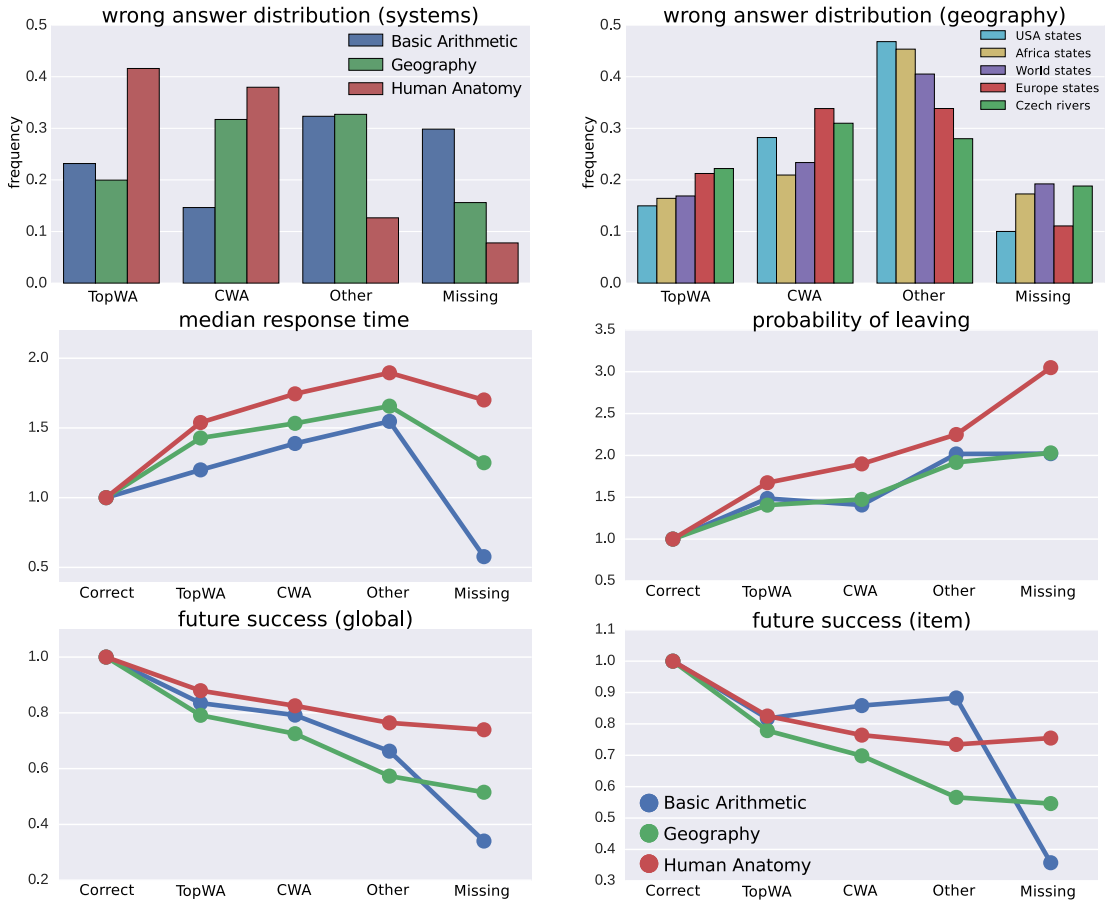


Figure 1: The first line shows frequency of different categories of wrong answers for different systems and for selected maps in geography system. The rest of the figure shows properties of different categories of answers normalized with respect to correct answers.

Figure 2: Left: A confusion graph for European countries (showing only the most significant edges). Right: Clustering of European countries based on community detection in the confusion graph.

5. REFERENCES

[1] Deanne M Adams, Bruce M McLaren, Kelley Durkin, Richard E Mayer, Bethany Rittle-Johnson, Seiji Isotani, and Martin van Velsen. Using erroneous examples to improve mathematics learning with a web-based tutoring system. *Computers in Human Behavior*, 36:401–411, 2014.

[2] Thomas Aynaud. Community detection for networkx, 2009.

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[4] John Seely Brown and Kurt VanLehn. Repair theory: A generative theory of bugs in procedural skills. *Cognitive science*, 4(4):379–426, 1980.

[5] Brian Butterworth, Noemi Marchesini, Luisa Girelli, and AJ Baroody. Basic multiplication combinations: Passive storage or dynamic reorganization? *The Development of Arithmetic Concepts and Skills: Constructive Adaptive Expertise*, pages 187–201, 2003.

[6] Michel C Desmarais and Ryan SJ d Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2012.

[7] Elena L Glassman, Jeremy Scott, Rishabh Singh, Philip J Guo, and Robert C Miller. Overcode: Visualizing variation in student solutions to programming problems at scale. *ACM Transactions on Computer-Human Interaction*, 22(2):7, 2015.

[8] George Goguadze, Sergey Sosnovsky, Seiji Isotani, and Bruce Mclaren. Evaluating a bayesian student model of decimal misconceptions. In *Educational Data Mining 2011*, 2010.

[9] Jonathan Huang, Chris Piech, Andy Nguyen, and Leonidas Guibas. Syntactic and functional variability of a million code submissions in a machine learning mooc. In *AIED 2013 Workshops Proceedings Volume*, page 25, 2013.

[10] Barry Peddycord Iii, Andrew Hicks, and Tiffany Barnes. Generating hints for programming problems using intermediate output. In *Educational Data Mining*, 2014.

[11] Seiji Isotani, Deanne Adams, Richard E Mayer, Kelley Durkin, Bethany Rittle-Johnson, and Bruce M McLaren. Can erroneous examples help middle-school students learn decimals? In *Towards Ubiquitous Learning*, pages 181–195. Springer, 2011.

[12] Jaclyn K Maass and Philip I Pavlik Jr. How spacing and variable retrieval practice affect the learning of statistics concepts. In *Artificial Intelligence in Education*, volume 9112 of *LNCS*, pages 247–256. Springer, 2015.

[13] Thomas S McTavish and Johann Ari Larusson. Labeling mathematical errors to reveal cognitive states. In *Open Learning and Teaching in Educational Communities*, pages 446–451. Springer, 2014.

[14] Andy Nguyen, Christopher Piech, Jonathan Huang, and Leonidas Guibas. Codewebs: scalable homework search for massive open online programming courses. In *Inter. conf. on World Wide Web*, pages 491–502. ACM, 2014.

[15] Jan Papoušek and Radek Pelánek. Impact of adaptive educational system behaviour on student motivation. In *Artificial Intelligence in Education*, volume 9112 of *LNCS*, pages 348–357, 2015.

[16] Jan Papoušek, Radek Pelánek, and Vít Stanislav. Adaptive practice of facts in domains with varied prior knowledge. In *Educational Data Mining (EDM)*, pages 6–13, 2014.

[17] Jan Papoušek, Radek Pelánek, and Vít Stanislav. Adaptive geography practice data set, 2015. http://www.fi.muni.cz/adaptivelearning/.

[18] Radek Pelánek and Petr Jarušek. Student modeling based on problem solving times. *International Journal of Artificial Intelligence in Education*, 25(4):493–519, 2015.

[19] Radek Pelánek, Jiří Řihák, and Jan Papoušek. Impact of data collection on interpretation and evaluation of student model. In *Learning Analytics & Knowledge*, pages 40–47. ACM, 2016.

[20] James A Reffel. Cued vs. free recall in long-term memory of the fifty united states. *Current Psychology*, 16(3-4):308–315, 1997.

[21] Kelly Rivers and Kenneth R Koedinger. Automatic generation of programming feedback: A data-driven approach. In *Workshop on AI-supported Education for Computer Science*, page 50, 2013.

[22] Hans-Jürgen Schmidt. Students' misconceptions–looking for a pattern. *Science education*, 81(2):123–135, 1997.

[23] John Stamper, Tiffany Barnes, Lorrie Lehmann, and Marvin Croy. The hint factory: Automatic generation of contextualized help for existing computer aided instruction. In *Intelligent Tutoring Systems Young Researchers Track*, pages 71–78, 2008.

[24] Marthe Straatemeier. *Math Garden: A new educational and scientific instrument.* PhD thesis, Universiteit van Amsterdam, Faculty of Social and Behavioural Sciences, 2014.

[25] Ilkay Sudas and Cemil Gokten. Cognitive maps of europe: geographical knowledge of turkish geography students. *European Journal of Geography*, 3(1):41–56, 2012.

[26] Dina Tirosh. Enhancing prospective teachers' knowledge of children's conceptions: The case of division of fractions. *Journal for Research in Mathematics Education*, pages 5–25, 2000.

[27] Tom Verguts and Wim Fias. Interacting neighbors: A connectionist model of retrieval in single-digit multiplication. *Memory & cognition*, 33(1):1–16, 2005.

[28] Jiří Řihák. Use of time information in models behind adaptive system for building fluency in mathematics. In *Educational Data Mining, Doctoral Consortium*, 2015.

[29] Yutao Wang, Neil T Heffernan, and Cristina Heffernan. Towards better affect detectors: effect of missing skills, class features and common wrong answers. In *Learning Analytics And Knowledge*, pages 31–35. ACM, 2015.

Using Inverse Planning for Personalized Feedback

Anna N. Rafferty
Department of Computer Science
Carleton College,
Northfield, MN 55057 USA
arafferty@carleton.edu

Rachel A. Jansen
Department of Psychology
University of California,
Berkeley, CA 94720 USA
racheljansen@berkeley.edu

Thomas L. Griffiths
Department of Psychology
University of California,
Berkeley, CA 94720 USA
tom_griffiths@berkeley.edu

ABSTRACT

An increasing number of automated models can make inferences about learners' understanding based on their problem solving choices in interactive educational technologies. One potential use of these models is to personalize feedback interventions. We investigate using the output of an inverse planning model to choose feedback activities for learners. The inverse planning model uses the patterns of how a learner solves algebraic equations to estimate her proficiency on several discrete skills. The personalized feedback then focuses on the skill which is least proficient and includes a combination of existing educational content and scaffolded practice. We experimentally tested the effectiveness of personalizing the feedback based on the algorithm's estimate compared to simply providing a random feedback activity. The results show that completing the feedback was associated with performance improvements from pre- to post-test, but that personalized feedback was not associated with reliably more improvement. However, participants who received feedback about a skill that was far from mastery did show reliably more improvement than those who received feedback about an already-mastered skill. This suggests that there is potential in using the inverse planning algorithm to provide more effective learning experiences.

1. INTRODUCTION

Cognitive models of people's learning are often useful for better understanding behavior and can highlight what a particular learner knows and where she may be struggling. There are also an increasing number of educational resources available for learning specific topics, such as online videos, which might be effective for remediating a learner's struggles. However, there can be challenges when trying to close the loop between estimating a model of what someone knows and creating interventions based on that model to address misunderstandings or gaps in knowledge. The model is not a perfect assessment, and many interventions may be effective for a particular learner, making it difficult to determine if personalizing the intervention is valuable. While there are a number of models that have been used to change the behavior of an educational technology, such as providing problems until mastery [2], there has been less of a focus on using models based on behaviors in more open-ended learning environments to guide feedback and remedial interventions in these settings.

We address the problem of closing the loop between a model-based assessment of a learner's algebra skills and the experience the learner has in a web-based algebra activity. The model was an inverse planning model for algebra, which provides an assessment of specific algebra skills based on the pattern of how someone solves equations. While the model provides a profile of what a person may misunderstand, suggesting that it could be used to guide feedback interventions, its estimates also have some error, meaning that it will not perfectly identify misunderstandings for every person. Additionally, the model's assessment is based on a collection of problem solutions, meaning the feedback must be targeted at an overall skill or misunderstanding rather than performance on a specific problem. This differs from many contexts where feedback is provided in interactive educational technologies, but has the potential to facilitate longer interventions about specific concepts or skills. This type of feedback could connect a learner with existing resources about particular concepts, since rather than assisting with a single question, the feedback is remediating a more abstract area of struggle. Thus, we explore how the model's assessment of understanding can be used to provide feedback to learners that targets their misunderstandings.

We investigate this question by designing feedback interventions for specific skills and experimentally testing how people's performance changes from pre- to post-test based on the intervention that they are given. The feedback interventions combined relevant content from existing sources and scaffolded opportunities for practicing a particular algebra skill. In an experiment, we compared performance for people who completed a feedback intervention based on the algorithm's estimate of their skills versus those who completed an intervention that was chosen randomly. We found that both groups showed significant performance improvements from pre- to post-test, but the two groups did not differ in their amount of improvement. However, completing feedback about a skill that one was less proficient in was reliably associated with more improvement than completing feedback about a skill that was near mastery. These results suggest that the algorithm's assessment may be used to di-

rectly improve the educational technology, although there are a number of subtleties in how to do this effectively.

2. BACKGROUND

There has been a great deal of previous work related to assessing student understanding and providing interactive feedback to improve understanding. In our work, we are most interested in techniques where a student's actions or choices are used as part of the assessment of understanding, such as in open-ended learning environments (OELEs). OELEs are often used in science education, as they can provide opportunities for students to generate and test their own hypotheses [6]. Educational data mining has been used to better understand what behaviors are associated with learning in some of these environments, such as Betty's Brain [4], and these environments may provide feedback to students about their progress (e.g., [12]). Data mining is also used in these environments for assessments of skills, especially those like experimentation that are more difficult to measure in other environments [3]. However, it is rarer for the data mining to be used directly to inform feedback to students, and the feedback that is provided is frequently in the form of a short hint or suggestion about what to do. In mathematics education, there exist several systems, such as the Cognitive Tutor [2], that maintain a model of student learning and use this to adapt instruction, such as providing more problems on an unmastered skill; typically these systems assess student knowledge based on final answers rather than on what actions are taken to generate a solution. In both the science and mathematics systems the type of adaptive feedback differs from our focus on providing a somewhat longer session of feedback focused on re-teaching a particular skill.

While formative feedback to learners is an effective way to improve understanding and help create a more integrated base of knowledge [13], the problem of determining what type of feedback will be most effective is an area of active research. Much of the previous work on feedback in mathematics tutors has focused on progressively more informative hints (e.g., [5]). More holistic information based on assessments of skills may be provided to students, such as when making a learner model "open" to the learner [1], but this is not necessarily paired with feedback or interventions to remediate understanding. Research about teachers' responses to student work in educational technologies has found that teachers may customize their instruction in a variety of ways to adjust to student misunderstandings [8]. Based on this work, we were interested in how more holistic feedback that focuses on a particular skill that a student is struggling with, rather than a specific problem, might affect learning.

3. INVERSE PLANNING

In order to get a holistic assessment of a learner's algebra skills based on observing their behavior, we used a Bayesian inverse planning approach [11]. Bayesian inverse planning takes as input a set of step-by-step actions from a learner, and outputs a posterior distribution over possible levels of proficiency for various skills. This approach allows us to interpret people's patterns of behaviors while they solve algebraic equations in a relatively freeform interface. In this interface, shown in Figure 1, learners have the ability to enter step-by-step solutions to equations, with no constraints on whether individual steps are correct before entering a new

Figure 1: A screenshot of the step-by-step interface for solving algebraic equations. The user may solve the problem using any steps she chooses and record them in the interface.

step. The Bayesian inverse planning algorithm uses both the mathematical correctness of each step and the way it moves the learner towards the solution to diagnose proficiency; the model is substantially similar to that described in [10]. We provide a brief overview of the algorithm and its underlying model of problem solving.

Bayesian inverse planning is based on a generative model: it models how likely a person would be to choose each possible solution step if she had a particular understanding of algebra, and then uses this model to infer what understanding is most likely to have resulted in the observed solutions. To create this generative model, we need to specify how choices about solution steps are made as well as specifying the representation of possible understandings. Inverse planning treats algebraic equation solving as a Markov decision process (MDP), in which people choose actions to bring them closer to the goal of solving an equation with as few steps as possible. With each action, the person moves from one (partially solved) equation to another. In an MDP, the value $Q_h(s, a)$ of taking an action a given that the current equation is s can be approximated using dynamic programming. This long-term value is dependent on the person's understanding of algebra, denoted as h, since that understanding may change what actions she believes are possible or what next equation she generates from the current equation. We model people as following a noisy optimal policy when choosing actions: $p(a|s) \propto \exp(\beta \cdot Q_h(s, a))$, where β controls the level of noise. Intuitively, this policy assumes people tend to choose actions that they think will help them solve the problem efficiently but they do not do so deterministically. The parameter β is estimated for each individual, as described below.

In this model, understanding is represented by the level of proficiency for several skills. For each skill, the proficiency indicates whether the person generally applies the skill correctly or if she makes a particular type of error. The different levels of proficiencies form a *hypothesis space* of possible algebra understandings. The hypothesis space was based on past education and psychology research and consists of parameters for six skills (see [10] for details): moving terms, dividing by the coefficient of a term, applying the distributive property, combining terms, arithmetic, and planning. The first four parameters relate to specific rules of manipulating algebraic equations, while the latter two apply more broadly.

Each of the four algebra-specific parameters indicates whether the person is prone to a particular type of error or "mal-

rule" [9]. For moving terms, the mal-rule is failing to flip the sign of a term when moving it from one side of the equation to another; the inferred parameter is the probability of not following this mal-rule when moving a term. For dividing by the coefficient of a term, the mal-rule is multiplying rather than dividing (i.e., not using the reciprocal), and for applying the distributive property, the mal-rule is only distributing the coefficient to the first term rather than all terms. Both of these parameters, like moving terms, are probabilities. For combining terms, the mal-rule is combining unlike terms, such as a variable and a constant. This parameter is binary: the person either considers combining unlike terms when choosing actions or she does not.

The final two parameters for the hypothesis space are the arithmetic parameter and the planning parameter. The arithmetic parameter is the probability that a person accurately computes a calculation. The planning parameter is the parameter β in the noisily optimal policy: higher values for this parameter indicate very high probability of choosing the most efficient action for moving towards a solution, while values close to zero indicate very different choices from those expected by the model, such as choosing an action that does not make progress towards the solution or giving up prior to reaching a solution. This parameter is the only parameter not targeted for feedback, as a mixture of cognitive and motivational feedback might be most effective for improving planning and lessening the rate of non-answers.

The parameters above form a six-dimensional, continuous hypothesis space $\mathcal{H}$, where each point in the space represents one possible set of skill proficiencies h. Given this hypothesis space, the posterior distribution after observing the person's problem solutions D is calculated using Bayes' rule: for each $h \in \mathcal{H}$, $p(h|d) \propto p(h) \prod_{d \in D} p(d|h)$, where $p(h)$ is the prior distribution over the hypothesis space and $p(d|h)$ is the likelihood that the person would produce the observed step-by-step solution if she had the skill levels indicated in h. The prior favors higher levels of proficiency; intuitively, this means that the algorithm favors the part of the hypothesis space indicating normative algebra understanding unless it observes evidence in the solutions that non-normative steps are being taken. Because the hypothesis space is continuous, the posterior distribution cannot be calculated exactly. Instead, Markov chain Monte Carlo (MCMC) methods are used to compute an approximate posterior distribution. As shown in Figure 2, the resulting posterior distribution indicates both the most likely proficiency for each skill as well as the algorithm's confidence. In the figure, both the parameter for moving terms and the distributive property are close to one, but the estimate for moving terms is more certain; there is also a lower estimated proficiency for arithmetic than for the other skills. In order to use the posterior distribution for feedback, we calculate the mean value of the posterior on each skill dimension (shown as green lines in Figure 2).

4. FEEDBACK DESIGN

Given the output of the inverse planning algorithm, our goal was to "close the loop" by providing learners with a feedback activity that could help to remediate their understanding of a particular skill. In an attempt to minimize differences in feedback effectiveness due to quality rather than topic, all of the feedback interventions followed the same pattern. First,

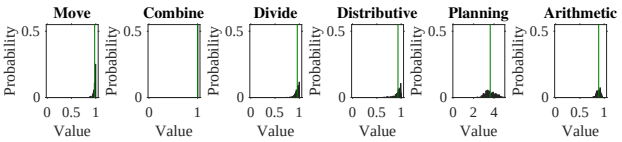


Figure 2: The inverse planning algorithm's assessment for a learner from the experiment. Each plot shows the posterior distribution for one skill. Larger values are closer to mastery.

an overview screen showed the learner two skills: the skill closest to mastery and the skill she would receive feedback about. In both cases, she was shown her proficiency level as a colored bar and a short description of the skill was provided. The bottom of the page told her that she would be learning more about the second skill that was shown; we refer to this skill as the *feedback skill*. On the next page, learners were shown a 2–5 minute embedded video about the feedback skill. Since there already exist a large number of freely availably educational videos, we aimed to connect learners to a relevant resource rather than create new tutorial content. All videos were sourced from Khan Academy[1], and were chosen because they targeted one of the five skills.

After the video, several stages of scaffolded practice were provided. For the four skills related to algebraic rules, the scaffolded practice began with four problems to highlight the core skill being practiced. For example, only the feedback focused on correctly applying the distributive property included practice on the distributive property. For these problems, the learner's steps were checked for correctness with each new step. If a mathematical error was detected, the step was highlighted and she was asked to fix it before continuing. After each problem, the learner was told the correct answer. Following these problems, eight problems were provided that still focused on the feedback skill, but checking of correctness was only provided after the learned submitted her answer. At that point, steps with errors were highlighted and the learner was given the opportunity to review them before continuing. These problems thus targeted the feedback skill, but included slightly less immediate assistance than the first set of problems. For the feedback targeting arithmetic, all twelve practice problems were arithmetic computations to complete rather than algebraic equations. Finally, all feedback finished with twelve algebra problems that were not specialized based on the feedback skill, with the intention for people to practice in context what they had learned from the skill-specific problems. The interface for these problems was the same as when doing general problem solving on the website: people had the opportunity to enter individual problem steps, and they were told whether they were correct before moving to the next problem.

5. EXPERIMENT

When we designed the feedback, our goal was to personalize what feedback someone was given based on the algorithm's assessment of their skills by assigning the person to complete feedback on their least proficient skill. While it is intuitively plausible that personalized feedback based on

[1]http://www.khanacademy.org/

this assessment might be more helpful than non-personalized feedback, there are several reasons to be skeptical. First, the algorithm's diagnosis is an approximation: there is error both in the MCMC estimate, and in the model itself. In general, the algorithm can interpret most problem solutions [10], but some people's behavior may be poorly fit by the model, resulting in poor accuracy for an individual. Additionally, the algorithm does not account for learning within the period that the skills are being assessed and depending on the person's behavior, there may be some skills about which we have very limited information. For example, a person might solve only a few problems using the distributive property, giving a relatively large confidence interval for possible skill proficiencies. A second concern about personalizing feedback is that learners who are struggling may be struggling in many skills. In that case, it may be that the personalization is unnecessary: most students who benefit from one feedback activity would also benefit from any of the other feedback activities. Thus, we ran an experiment to test whether the feedback activities were associated with learning and to examine whether personalized feedback produced larger learning gains than feedback that was not personalized based on the algorithm's assessment.

5.1 Methods

Participants. 200 participants in the USA were recruited from Amazon's Mechanical Turk (AMT) and compensated $4 for session 1, $6 for session 2, and $8 for session 3. Participants had taken an algebra course and had not completed college math classes beyond algebra.

Stimuli. Participants completed a multiple-choice assessment, solved algebra problems on a website, and responded to several surveys. The twelve question multiple-choice assessment was based on College Board ACCUPLACER® tests used for math placement in many postsecondary institutions[7]. The questions were substantially similar to the Elementary Algebra questions used in [11], but the numbers were changed to create two versions of the assessment.

All problem solving on the website used a similar interface to that shown in Figure 1. In sessions 1 and 3, learners were told whether or not they were correct immediately after submitting a problem. During the feedback in session 2, the interface behaved as described in the previous section.

In the surveys, participants indicated their demographics as well as prior math class experience. They also completed 18 questions focused on the usability of the website and the perceived helpfulness of the feedback.

Procedure. Participants completed three sessions, separated by at least one day. In the first session, all participants solved 24 equations on the algebra website, followed by the multiple-choice questions about elementary algebra topics. The website included a short tutorial about how to use the interface, and the 24 problems were generated based on templates. For example, one template was a constant plus a variable equal to a constant. The constants and coefficients for variables were generated randomly, but all participants shared the same templates. After a participant completed all problems on the website, the diagnosis for that participant was computed automatically by the inverse planning algorithm, and results were stored in the database for the participant's next session. Participants were randomly assigned to receive version one of the multiple-choice questions or version two; these versions were identical except for changes to the exact numbers used in the problems.

In the second session, participants completed one of the feedback activities. They were randomly assigned to either *targeted* or *random* feedback. Those receiving targeted feedback completed the feedback activity for the skill which the algorithm estimated they had least proficiency; those receiving random feedback completed one of the five feedback activities selected uniformly at random.

In the third session, participants again solved 24 equations on the algebra website, followed by the multiple-choice questions about elementary algebra topics. Just as in the first session, participants all completed problems on the website that used the same templates. For the multiple-choice questions, each participant completed the version of the questions that they did not complete in the first session. Finally, participants ended the third session by completing the demographics and usability surveys.

5.2 Results

82% of participants completed all three parts of the experiment in a single session. Several participants were removed due to technical problems, such as needing to restart the computer during a session and thus losing their place in the activity. The results that follow include only the 164 participants who completed all parts of the experiment.

Responses to our demographics questions suggest that participants came in with varying levels of mathematics background and that for most, significant time had passed since they had last studied algebra in school. 98% of participants reported what previous math classes they had taken, in college or in high school. 62% of those who responded had taken no math classes beyond geometry (typically at a high school level); the remaining participants had taken trigonometry, pre-calculus, or calculus at a high school level. A number of participants who reported taking one of these higher-level courses in high school also reported taking a college algebra class. Thus, we would expect all participants to have prior experience with solving equations, but to be likely to have some gaps in their knowledge.

We first examined changes in participants' performance between the first session, before getting feedback, and the final session, after getting feedback. Results from the first session confirmed that participants were on average far from ceiling on the task: they correctly answered an average of 7.2 multiple-choice questions out of a total of 12, and correctly answered an average of 12.4 out of the 24 algebra problems on the website. There was a small increase in the number of multiple-choice questions answered correctly in the final session. Using a repeated-measures ANOVA with factors for time of test, condition, and a random factor for participant, we found that this main effect was reliable ($F(162, 1) = 15.7$, $p < .001$), but there was no interaction between condition (targeted versus random feedback) and time of test. Given that many of the questions focused on skills that were not directly targeted by our intervention, in-

cluding some quadratic equations and linear inequalities, it is not surprising that we see only a small improvement from the first to the final session. The increase in performance was somewhat larger for the algebra equations solved on the website: participants correctly answered 23% more problems correctly, for a mean of 16.6 problems correct in the final session. We again used a repeated-measures ANOVA with factors for time of test, condition, and a random factor for participant to analyze the reliability of this finding, and found that there was a main effect for time of test ($F(162,1) = 89.9$, $p < .001$), but no interaction between time of test and condition.

To better understand why there was no interaction between condition and the amount of improvement, we examined the estimated proficiency level of the skills for which feedback was given. On average, the targeted condition selected skills that had lower levels of proficiency (average proficiency level of 0.56 versus 0.88; $t(162) = 7.03$, $p < .001$), indicating that in many cases, there were large differences between the least mastered skill and a random skill. However, there were a number of participants in the random condition who received feedback about a skill with which they were struggling as well as participants in the targeted condition who did not have any skills that were far from mastered. To test whether participants who received feedback that was more appropriate for them improved more than participants who received feedback that was less appropriate for them, we divided all participants into two categories: those who received feedback about a skill that was estimated to be less than a proficiency level of 0.85 (an *unmastered* skill) and those who received feedback about a skill that was at a proficiency level greater than or equal to 0.85 (a *mastered* skill). This criterion categorizes 46% of participants as receiving feedback about an unmastered skill. As shown in Figure 3, participants who received feedback about an unmastered skill improved more than those who received feedback about a mastered skill. A repeated-measures ANOVA with factors for whether the feedback skill was already mastered, time of test, and a random factor for the participant showed that there was a main effect of time of test as well as an interaction between time of test and whether the feedback skill was already mastered ($F(162,1) = 9.42$, $p < .01$). To ensure that this result was not simply due to the cutoff level we chose for mastery, we also examined a categorization based on mastery level 0.9, and found the same trends ($F(162,1) = 46$, $p < .05$). While these results must be interpreted with some caution, as participants were not randomly assigned to the two categories, they suggest that receiving feedback that the algorithm indicates is more appropriate can result in greater improvements in performance.

Based on the fact that proficiency level influenced the effectiveness of the feedback, we examined the distribution of proficiencies for individual participants. We were interested in whether participants tended to have all skills at a similar level or whether they usually had some skills that were mastered and some that were unmastered. As shown in Figure 4, 35% of participants were at mastery for all skills, where mastery is defined as proficiency of at least 0.85, and 14% of participants were not at mastery for any skills. The remaining 51% of participants who had some mastered skills and some unmastered skills are arguably those that might most benefit from targeted rather than random feedback. A repeated-measures ANOVA with factors for time of test, condition, and a random factor for participant shows that there is a significant interaction between time of test and condition when restricting the data to these participants: as shown in Figure 5, those who completed targeted feedback improved almost twice as much those who completed random feedback (average improvement 5.3 versus 3.0; $F(82,1) = 5.64$, $p < .05$).[2] This suggests that inverse planning can provide a benefit for these participants: it allows us to determine what skill(s) will be appropriate targets for feedback.

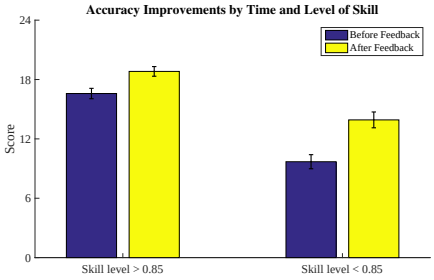


Figure 3: Improvement from first to last session in accuracy on website problems, categorizing participants based on prior level of proficiency in feedback skill. Participants who received feedback about an unmastered skill improved more from the first to the final session than those who received feedback about a mastered skill.

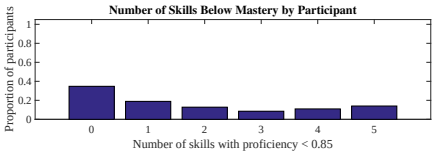


Figure 4: Count of the number of unmastered skills by participant.

6. DISCUSSION

Our goal in the feedback design and the experiment was to evaluate the benefit of connecting the holistic assessment and the feedback activities. While many of the feedback problems provided practice on multiple skills, since multiple skills are required to solve the algebraic equations, there was specialization in our feedback based on the algorithm's assessment. Our results show that overall, participants' performance improved after completing the feedback activities. The effects of personalization on the size of this effect were mixed: across all participants, feedback targeted at someone's weakest skill was not associated with reliably more improvement than feedback about a random skill, but restricted to those who had some mastered and some unmastered skills, we observed more improvement for those receiving the targeted feedback compared to those receiving the random feedback. This suggests that there is promise in using the inverse planning algorithm's assessment to connect

[2]With mastery level set at 0.9, this effect is marginally significant (average improvement 4.2 versus 2.7; $F(103,1) = 3.33$, $p = .07$).

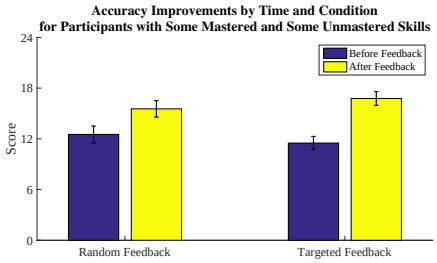


Figure 5: Improvement from first to last session in accuracy on website problems, restricted to participants with some unmastered and some mastered skills. Participants show reliable improvement, and participants who received targeted feedback tended to improve more than those who received random feedback.

learners to relevant resources and personalize feedback activities, although further investigation is needed to determine ways to make this personalization even more effective.

There are several limitations of this work. First, our population of AMT workers may not be typical of algebra learners. These people were paid to participate in the study, and may differ in motivation and background from those who would use the website by choice. However, their varied backgrounds may be typical of adult learners who are trying to surmount barriers such as algebra at the community college level, a group we are particularly interested in reaching. Second, this experiment does not separate whether the content of a feedback intervention is helpful from whether the targeting of that feedback is accurate. We intend to further evaluate these two components to better understand what the maximum benefit of this type of feedback would be if targeting was perfectly accurate, but any evaluation of the overall effectiveness of the knowledge diagnosis-feedback loop must acknowledge that inaccuracies in the diagnosis may lead to the personalization being less effective.

In future work, there are a number of ways we will explore how to design more effective personalized feedback and investigate variations in how to use the algorithm for personalization. Our intervention was relatively short, with most participants taking about an hour for the session in which feedback was provided. One might expect the effects of personalization to be cumulative, with targeted feedback being most helpful when learning over a longer period; in that case, the targeting could be used to remediate the same skill multiple times if struggles were still evident or to recognize that say, one session of feedback had resulted in several skills reaching mastery and skipping the already mastered skills. Such longer interventions are likely to have larger effects, and may highlight whether targeted feedback is overall more effective or whether there is a subset of participants for which targeting makes a difference. Another area to explore is providing the profile generated by the inverse planning algorithm to the learner and using this in conjunction with targeted feedback, random feedback, or feedback chosen by the learner. The current system provides learners with the algorithm's assessment of several of their skills, but it does not allow them to make choices about what feedback they receive. Choice might be useful for those not well-modeled by the algorithm or in cases where several non-mastered skills have been identified; however, it is also possible that struggling learners are unable to understand the possible types of feedback in order to make a good choice. Finally, there are several ways we might adjust how the algorithm's output is linked to feedback. The diagnosis includes information about the algorithm's certainty. This might be used to focus on skills that we are confident are unmastered. Additionally, the algorithm outputs a diagnosis of planning efficiency, but this was not used for feedback. Low levels of this parameter can be indicative of someone who frequently gives up or who is not well fit by the model. In either situation, it may not be appropriate to simply give feedback about the least proficient skill. Overall, the results in this paper serve as first steps for a larger investigation into how to effectively close the loop between holistic assessments of misunderstandings and guiding personalized feedback interventions for learners.

Acknowledgements. This work was funded by NSF grant number DRL-1420732 to Thomas L. Griffiths. Thanks go to Jonathan Brodie and Sam Vinitsky for programming parts of the feedback.

7. REFERENCES

[1] S. Bull and J. Kay. Student models that invite the learner in: The SMILI open learner modelling framework. *International Journal of Artificial Intelligence in Education*, 17(2):89–120, 2007.

[2] A. T. Corbett, K. R. Koedinger, and W. Hadley. *Cognitive tutors: From the research classroom to all classrooms*, pages 235–263. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, 2001.

[3] J. D. Gobert, M. Sao Pedro, J. Raziuddin, and R. S. Baker. From log files to assessment metrics: Measuring students' science inquiry skills using educational data mining. *Journal of the Learning Sciences*, 22(4):521–563, 2013.

[4] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *JEDM-Journal of Educational Data Mining*, 5(1):190–219, 2013.

[5] K. R. Koedinger and V. Aleven. Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, 19(3):239–264, 2007.

[6] S. M. Land. Cognitive requirements for learning with open-ended learning environments. *Educational Technology Research and Development*, 48(3):61–78, 2000.

[7] K. D. Mattern and S. Packman. Predictive validity of accuplacer scores for course placement: A meta-analysis. Technical report, College Board, December 2000.

[8] C. F. Matuk, M. C. Linn, and B.-S. Eylon. Technology to support teachers using evidence from student work to customize technology-enhanced inquiry units. *Instructional Science*, 43(2):229–257, 2015.

[9] S. Payne and H. Squibb. Algebra mal-rules and cognitive accounts of error. *Cognitive Science*, 14(3):445–481, 1990.

[10] A. N. Rafferty and T. L. Griffiths. Interpreting freeform equation solving. In *Artificial Intelligence in Education*, pages 387–397. Springer International Publishing, 2015.

[11] A. N. Rafferty, M. M. LaMar, and T. L. Griffiths. Inferring learners' knowledge from their actions. *Cognitive Science*, 39(3):584–618, 2015.

[12] J. R. Segedy, J. S. Kinnebrew, and G. Biswas. The effect of contextualized conversational feedback in a complex open-ended learning environment. *Educational Technology Research and Development*, 61(1):71–89, 2013.

[13] V. Shute. Focus on formative feedback. *Review of Educational Research*, 78(1):153–189, 2008.

Pattern mining uncovers social prompts of conceptual learning with physical and virtual representations

Martina A. Rau
Department of Educational Psychology
University of Wisconsin—Madison
1025 W. Johnson St
Madison, WI 53706
+1-608-262-0833
marau@wisc.edu

ABSTRACT

To succeed in STEM, students need to connect visual representations to domain-relevant concepts, which is a difficult task for them. Prior research shows that physical representations (that students manipulate with their hands) and virtual representations (that they manipulate on a computer) have complementary advantages for conceptual learning. Further, physical and virtual representations are often embedded into different social classroom practices. Thus, to optimally combine these representation modes, we need to understand what social events prompt students to connect representations to concepts, and if different representation modes afford different social prompts. A multiple-case study with 12 high-school students addresses this question. Student pairs worked with physical and virtual representations of chemistry. Frequent patterns obtained from discourse data show that students incrementally co-construct concept-representation connections, and that instructor prompts are key triggers of these connections for both representation modes. Meta-cognitive statements serve as important prompts in the absence of an instructor when students work with virtual representations. I discuss implications for interventions that combine physical and virtual representations.

Keywords

Physical and virtual representations, educational technology, collaboration, conceptual and social learning processes, STEM.

1. INTRODUCTION

Novice students in science, technology, engineering, and math (STEM) domains grapple with a *representation dilemma* [1]: they have to use visual representations they have never seen before to make sense of concepts they have not yet learned. Educators often take for granted that students can see meaningful concepts in representations [2]. However, much evidence shows that students struggle in connecting concepts to visual representations [3]. Their failure to make such concept-representation connections can impede their learning [4]. For example, in chemistry, difficulties in making concept-representation connections affect students' understanding of key concepts related to atomic structure and bonding [5]. This issue applies to most STEM domains: because many key concepts cannot be directly observed, STEM domains heavily rely on visual representations [3]. Thus, STEM instruction typically provides conceptual prompts to help students make concept-representation connections [6].

Research in many STEM domains—including chemistry—shows that different *representation modes* provide different types of prompts for concept-representation connections [7]. *Physical representations* are tangible objects that students manipulate with their hands (Figure 1, top). In physical representations, haptic sensory input, experiences of movement, and continuous changes serve as prompts by making concepts intuitively accessible [7, 8]. By contrast, *virtual representations* are digital visualizations that students manipulate via mouse or text input (Figure 1, bottom). In virtual representations, visualizations and manipulations of invisible processes and immediate feedback can serve as prompts for concept-representation connections [7]. Thus, physical and virtual representations serve complementary roles in prompting for students to make concept-representation-connections [7, 9].

Besides providing different types of conceptual prompts for concept-representation connections, physical and virtual representations may provide different types of *social prompts*. Social prompts are discourse events that elicit collaborative co-construction of such connections [10]. Such events can emerge from student-student or student-instructor interactions. Because *physical representations* are typically used in collaborative contexts, interactions among students and instructors may prompt concept-representation connections [11]. By contrast, *virtual representations* are embedded in educational technologies that provide help in making concept-representation connections. In this context, students may work individually or collaboratively, typically with less help from an instructor [12]. Hence, interactions with instructors may be less important in prompting concept-representation connections. Thus, because physical and virtual representations are embedded in different social classroom practices, they may yield different social prompts for concept-representation connections.

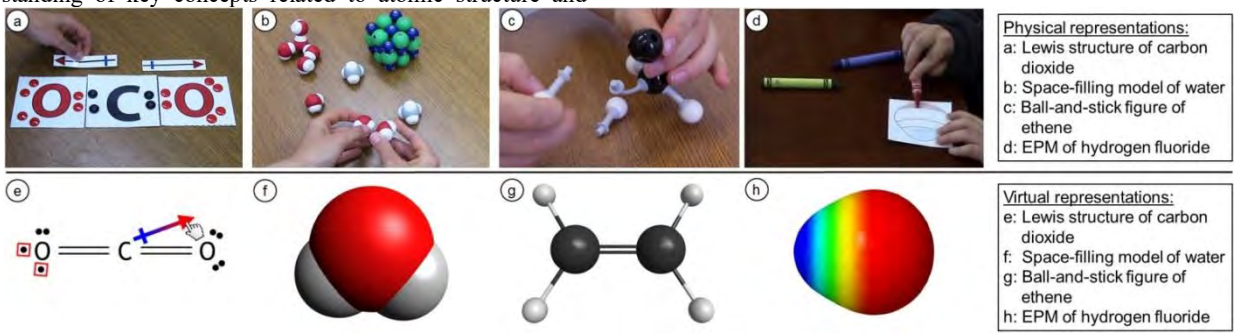


Figure 1. Physical representations (top) and virtual representations (bottom) of chemical molecules

Considering what social events serve as prompts for concept-representation connections is important for the design of instructional interventions that combine physical and virtual representations. Prior research has not investigated whether different representation modes afford different types of social prompts for concept-representation connections. At a theoretical level, addressing this question will help us understand the mechanisms by which representation modes affect students' ability to make concept-representation connections. It will also help us understand why one representation mode may be more effective than another for a given concept. At a practical level, it will allow us to design instructional activities that take advantage of the social prompts that the different representation modes afford.

The goal of this paper is to take a first step towards identifying social prompts of concept-representation connections for physical and virtual representation modes. To this end, I used a multiple-case study approach; specifically, I observed and recorded collaborative discourse among six student pairs over an extended learning period. Case-study approaches are particularly appropriate for investigating how social processes unfold over a longer learning intervention within the given social classroom practices [13]. The study compared two instructional contexts: (1) student pairs working with physical representations while receiving support from an instructor and (2) student pairs working with virtual representations embedded in an educational technology.

To identify social prompts of concept-representation connections, I applied frequent pattern mining to discourse data. This analysis identified social prompts that are successful for both representation modes and social prompts that were specific to a particular representation mode. I discuss implications for blending interventions that combine physical and virtual representations.

2. METHODS

2.1 Multiple-Case Study

Participants were 12 students from a small charter high school in the Midwestern U.S. The study was conducted as part of a chemistry workshop. Students had very limited prior knowledge about the concepts and the visual representations. The study took place as part of an in-school workshop on 3 days spread across 4 weeks. Each study day was 3h long. Prior to day 1, the teacher gave an introduction on chemical bonding. On day 1, students received an introduction into collaborative strategies and then worked on the chemistry workshop materials for the remaining study days.

All students were randomly assigned to pairs for the duration of the study. For each study day, the pairs were randomly assigned to a sequence of representation mode (i.e., physical-then-virtual, virtual-then-physical). For example, a pair might be assigned to the physical-then-virtual order for day 1. This pair would work with physical representations for the first half of day 1 and then switch to virtual representations for the second half of day 1. On day 2, the pair was randomly assigned to a new sequence.

The workshop covered basic concepts related to the polarity of chemical bonds. Students were presented with the visual representations shown in Figure 1: Lewis structures, ball-and-stick models, space-filling models, and electrostatic potential maps. Each was presented in the physical and virtual mode. When working with *physical representations*, students received a worksheet that asked them to construct a physical representation of a molecule, answer questions about the target concepts (e.g., about electronegativity) and about how the representation depicts these concepts. Each student pair was teamed up with an instructor—a research assistant who was trained on facilitating student collaboration and on the chemistry concepts covered. Instructors provided feedback and assistance as students solved the problems.

*Virtual representation*s were integrated in an educational technology for chemistry: Chem Tutor [14]; a type of intelligent tutoring system designed specifically to help students make concept-representation connections. To this end, Chem Tutor provides interactive virtual representations that students manipulate to solve problems about bonding. Chem Tutor prompts students to reflect on how each visual representation depicts particular concepts. Chem Tutor provides error-specific feedback and hints on demand. Chem Tutor was shown to significantly enhance learning of chemistry knowledge and conceptual understanding of representations [14]. While working with Chem Tutor, students could request help from an instructor who circulated the classroom.

2.2 Analysis

The goal of the analysis was to identify social events that prompt students' concept-representation connections and to investigate whether these prompts differ between representation modes.

The first step in the analysis was to code discourse data. All interactions among students and instructors were video-taped and transcribed. To develop a coding scheme, we used a grounded, bottom-up approach: we summarized discourse utterance-by-utterance to discover emerging themes. Next, we formalized these themes as codes, and then applied the codes to the discourse data. The coding scheme comprises 45 codes (see Table 1 for examples). Inter-rater reliability was substantial with kappa = .77.

The second step in the analysis was to identify discourse segments in which students succeed in making a concept-representation connection, defined as establishing the relation between a visual feature in a representation and the domain-relevant concept it illustrates [6]. Hence, a concept-representation connection was operationalized as an utterance made by a student that correctly refers to a concept and a representation (e.g., Table 2, #5).

The third step in the analysis was to operationalize social events that may prompt students to make concept-representation connections. In principle, any aspect of student-student or instructor-student discourse could serve as a social prompt: mentioning a concept, encouragement, evaluating, a meta-cognitive statement, a mistake, etc. Hence, I considered any code as a potential prompt.

The fourth step in the analysis was to specify the unit of analysis. Because I was interested in *social* events as prompts, I defined two consecutive discourse turns as the unit of analysis (i.e., utterances by two different speakers). I segmented the discourse data in the following way. First, I identified turns with concept-representation connections (e.g., Table 2, row 5). Second, I identified the two prior turns and considered them as a case (e.g., rows 3-4 in Table 2). This case was labeled as 'connection present' (i.e., a concept-representation connection occurs in the next turn). Third, I segmented the remaining discourse data such that two consecutive turns serve as a case (e.g., rows 1-2 in Table 2), labeled as 'connection absent' (i.e., no concept-representation connection in the next turn). Thus, each case was composed of two consecutive turns, labeled as connection-present/absent, annotated with codes, speaker (student or instructor) and mode (physical or virtual). Table 3 shows an overview of the dataset.

The final step in the analysis was to search for social events that trigger concept-representation connections. Given the focus on social mechanisms, I was interested in discovering which codes co-occur in collaborative discourse. To this end, I used frequent pattern mining to identify undirected patterns that describe which

Table 1. Subset of codes in the coding scheme with examples from the transcripts.

Code	Definition	Example
Concept	Utterances that relate something to a scientific concept	"They want to be able to make a complete number, a complete number of the eight on the outside"
Concept-request	Suggesting / prompting utterances that relate something to a concept	"What's the rule for the bonding?"
Representation	Utterances that relate something to the representation; utterances that explain information shown by a representation	[pointing at a representation] "So, one, two, three, four, five. He have five."; [pointing at a representation] "So, wait, that's carbon?"
Representation-request	Suggesting / prompting utterances that relate something to the representation; utterances that explain information shown by a representation	"By looking at the Lewis structure, can you answer the question about electronegativity?"; "What are these things [points at dots in Lewis structure]?"
Assent	Expression of approval or agreement	"yeah"; "ok"; "I know."; "Mmhmm."
Meta-confusion	Utterances about oneself that describe confusion about how to proceed or about a concept, or about not knowing a concept	I don't know."; "this is very confusing." "Maybe."; "This is hard."; "So, now we're stuck."; "I don't get it why it's lines."
Meta-understanding	Utterances about oneself that describe a novel insights or understanding of how to proceed or of a concept	"Got it "; "Well, I know that part"; "I like this explanation."; "then I was like, well, duh"; "We've been making this so much harder than it is!"
Reading	Reading the problems statement or instructions or hints / feedback from Chem Tutor	"well it says right here that, "Choose the letters that show each atom,"
Explanation	Utterances that explain / elaborate a concept	"But when they say dinitrogen, means they bonded."; "I'll give a little bit more help."; "So, carbon has more electrons than hydrogen."
Explanation-request	Suggesting / prompting utterances that explain / elaborate a concept	"So what do you think that that is?"; "Could you try, try to put as a complete sentence"; "But why?"; "How did you know?"
Metaphor	Utterances that use a metaphor, intuitive example, embellished language to describe an abstract concept	"To make it lock on kind of."; "can I borrow your electrons"; "It's the same pulling forces."; "So, like magnetic, plus and minus.";

Table 2. Excerpt transcript showing 4 turns before a concept-representation connection (*turn #5*), with codes assigned to each turn. All student names are fake.

#	Speaker	Utterance	Codes
1	Brigid	Electronegativity are the same so makes it covalent which is no difference.	Concept
2	Adriana	[reads] Does the Lewis structure show the polarity? Why or why not? Um. I'd say- I feel like no, be- Well, yeah. I don't know.	Reading; meta-confusion
3	Brigid	What does polarity mean?	Explanation-request; concept-request
4	Instructor	Polarity means plus and minus. Polarity means- This [points at representation] By looking at this one, can you see it has like electronegativity or stuff. Polarity means that-	Explanation; metaphor; representation-request; concept-request
5	Adriana	I mean, like yeah, it doesn't like show really like the pulling or the not pulling or the same.	Explanation; ***representation; concept;*** metaphor

codes often occur together [15, 16]. I ran this algorithm separately for cases with connections present or absent and for physical and virtual representations. Essentially, this analysis discovered:

1. Frequent patterns for cases with concept-representation connections *present* for *physical* representations
2. Frequent patterns for cases with concept-representation connections *absent* for *physical* representations
3. Frequent patterns for cases with concept-representation connections *present* for *virtual* representations
4. Frequent patterns for cases with concept-representation connections *absent* for *virtual* representations

Comparing findings 1 and 2 identified prompts of concept-representation connections for physical representations. Comparing findings 3 and 4 identified prompts of concept-representation connections for virtual representations. Comparing findings 1 and 3 identified differences between representation modes.

3. RESULTS

In the following, I first discuss which discourse patterns were found to prompt concept-representation connections with physical representations or with virtual representations. Then, I compare the physical and virtual representation modes.

3.1 Physical models

To identify prompts of concept-representation connections with physical representations, I considered patterns found only for cases with a *present* concept-representation connection (i.e., cases that correspond to two turns followed by a concept-representation connection). Table 4 shows statistics for the patterns.

Several results are worth noting. First, it stands out that all patterns involve either a reference to a concept or to a representation.

Table 3. Number of cases by representation mode and speaker.

Representation mode	Label				Speaker			
	Connection present		Connection absent		Student		Instructor	
Physical	229	(7.33%)	2,895	(92.67%)	2,115	(67.70%)	1,009	(32.30%)
Virtual	67	(3.28%)	1,976	(96.72%)	1,780	(86.13%)	263	(12.87%)

Table 4. Frequent patterns for physical representations (underlined: instructor utterances, italics: patterns that overlap with virtual representations).

Frequent pattern	Support	Confidence
1. instructor-assent; student-concept	*0.100*	*0.410*
2. instructor-assent; student-representation	*0.087*	*0.377*
3. instructor-representation-request; instructor-concept-request	*0.074*	*0.684*
4. student-representation; student-concept	*0.201*	*0.803*
5. instructor-assent; student-representation; student-concept	*0.083*	*0.536*

This finding suggests that it may be easiest for students to make a concept-representation connection if discourse is already focused on the concept or representation. A related finding is that 3 of 5 patterns include references to *both* concepts and representations—either as a request to relate to concepts and representations by the instructor (#3 in Table 4) or by the students themselves (#4 and #5). These patterns have the highest support and confidence. Hence, students may be particularly likely to make a concept-representation connection it already occurs in previous discourse.

Second, 4 of 5 patterns involve instructor utterances. This finding suggests that instructors may be better than students at prompting concept-representation connections.

Finally, 3 of 5 patterns include assent by the instructor. Assent is defined as agreement with a previous statement (see Table 1), often in the form of encouragement (e.g., "mhm). In the identified patterns, such encouragement co-occurs with references to a concept or to a representation (or both) provided by one of the students or by the instructor. This finding suggests that encouragement by the instructor—when discourse is already focused on a concept or representation—prompts students to elaborate by making a concept-representation connection.

3.2 Virtual models

To identify triggers of concept-representation connections with virtual representations, I considered patterns found only for cases with a *present* concept-representation connection. Table 5 shows statistics for these patterns.

The following findings stand out. First, all patterns include a reference to a concept or to a representation. Hence, students may be likely to make a concept-representation connection if discourse is already focused on a concept or on a representation. A related result is that 7 of 16 patterns include a reference to both concept and representation (either as request by the instructor, or a direct reference to both by the instructor or the student). These patterns

Table 5. Frequent patterns for virtual representations (underlined: instructor utterances, italics: overlap with physical representations).

Frequent pattern	Support	Confidence
1. instructor-assent; instructor-concept	0.075	0.420
2. student-metaConfusion; student-representation	0.104	0.393
3. student-metaUnderstanding; student-representation	0.075	0.471
4. student-metaUnderstanding; student-concept	0.075	0.476
5. student-metaConfusion; student-concept	0.075	0.386
6. student-concept; student-assent	0.134	0.388
7. student-representation; student-assent	0.134	0.378
8. instructor-concept-request; instructor-concept	0.060	0.468
9. instructor-representation-request; instructor-representation	0.060	0.468
10. instructor-representation-request; instructor-concept	0.060	0.508
11. student-assent; instructor-representation; instructor-concept	0.060	0.568
12. student-metaConfusion; student-representation; student-concept	0.075	0.468
13. instructor-representation-request; instructor-representation; instructor-concept	0.060	0.637
14. student-metaUnderstanding; student-concept; student-representation	0.060	0.463
15. student-assent; student-concept; student-representation	0.119	0.550
16. instructor-representation; student-assent	0.060	0.299
17. instructor-assent; student-concept	*0.090*	*0.374*
18. instructor-assent; student-representation	*0.104*	*0.428*
19. instructor-representation-request; instructor-concept-request	*0.075*	*0.714*
20. student-concept; student-representation	*0.254*	*0.792*
21. instructor-assent; student-representation; student-concept	*0.090*	*0.539*

had the highest support and confidence. Hence, students may be particularly likely to deepen their discussion about a connection if prior discourse already focuses on the connection.

Second, 7 of 16 patterns involve instructor utterances. This ratio seems surprisingly high, given that students worked without the instructor for most of the time. Recall that when working with virtual representations, instructor support was available only upon request, and that when students worked with virtual representations, they generated 86.13% of the utterances—instructors only 12.87% (see Table 2). Thus, this finding may indicate that students need help from an instructor to make concept-representation connections, even if they receive technology support.

Third, 6 of 16 patterns include assent by the instructor (4 of 6) or a student (2 of 6). Recall that assent is defined as agreement with a previous statement (see Table 1), often in the form of encouragement. Again, assent always co-occurs with a reference to a concept or representation. Hence, this finding suggests that encouragement can prompt a concept-representation connection—regardless of whether it is provided by a student or a tutor.

Fourth, 4 of the 7 patterns that involve instructor utterances involve explicit requests for the student to relate to a concept or a representation. This request is always combined with an instructor reference to a concept or to a representation. This finding suggests that prompts to elaborate on a previously mentioned concept or representation yields concept-representation connections.

Finally, 6 of 16 patterns include a meta-cognitive utterance by the student about understanding (3 of 6) or confusion (3 of 6). All of these meta-cognitive utterances co-occur with a reference to a concept and/or a representation. None of these meta-cognitive utterances co-occur with instructor utterances. This finding suggests that meta-cognitive statements about one's own understanding can prompt concept-representation connections; for example, after a student voices confusion about a concept, the partner may use a representation to explain the concept.

3.3 Comparing physical and virtual modes

Finally, I investigated whether prompts of concept-representation connections differ by representation mode. The following commonalities stand out. First, all patterns found for physical representations were also found for virtual representations. Hence, prompts that help students connect concepts to physical representations are also successful prompts for virtual representations.

Second, patterns with highest support and confidence for both representation modes involved relations to concepts and/or representations, indicating that students co-construct concept-representational competencies incrementally, over the course of consecutive social exchanges.

Third, the instructor plays a prominent role in prompting concept-representation connections both for physical and virtual representations: instructor utterances were involved in 4 of 5 patterns for virtual representations and in 7 of 16 patterns for physical representations. This result suggests that the role of an instructor is critical to students' success in making concept-representation connections, regardless of representation mode.

Fourth, assent that co-occurs with a reference to concepts or representations plays an important role for both representation modes. Hence, encouraging students to elaborate by agreeing with prior utterances may prompt concept-representation connections.

Several differences between representation modes stand out. First, students made fewer concept-representation connections with virtual representations (3.28%; see Table 2) than with physical representations (7.33%). Given the finding that instructors play a critical role for concept-representation connections, it may be that the lower involvement of an instructor when students work with virtual representations accounts for this difference.

Second, when students work with physical representations, assent seems to prompt concept-representation connections only when it is provided by the instructor. By contrast, when students work with virtual representations, assent provided by the student partner also prompts concept-representation connections. Hence, this type of prompt may be one that students can take responsibility for when working collaboratively without instructor support.

Finally, meta-cognitive utterances of confusion or understanding of concepts or representations were important prompts only for virtual representations. Given that none of the patterns that included meta-cognitive utterances included instructor utterances, it seems that meta-cognitive utterances are a major mechanism by which students can prompt concept-representation connections in the absence of instructor support.

4. DISCUSSION

My goal was to investigate the representation dilemma: how novice students make connections between new concepts and new representations. I investigated which social events in collaborative classroom practices prompt students' concept-representation connections. Using frequent pattern mining, I identified such prompts for physical and virtual representations.

A key finding was that prompts with the highest confidence and support contained relations to a previously mentioned concept or representation, regardless of representation mode. This finding suggests that the conceptual process by which students make concept-representation connections is mediated by a gradual, incremental social mechanism. Students may first discuss a concept or a representation separately from one another before they negotiate the connection between the two.

A further finding was that instructors played a crucial role in prompting concept-representation connections, regardless of the representation mode. With respect to physical representations, this finding is not surprising because students have no other way of receiving feedback and assistance. However, with respect to virtual representations, this finding is surprising because the representations were embedded in an educational technology that supported concept-representation connections (and was shown to be successful in doing so [14]). Hence technology support for concept-representation connections may not be able to "replace" instructor support—at least when students have little prior knowledge about the concepts or representations.

Finally, the results showed that meta-cognitive statements can prompt concept-representation connections when students work on virtual representations. Meta-cognitive statements were the only successful prompts when an instructor was not involved. The social mechanism underlying this effect may be that a meta-cognitive statement by one student prompts the other to explain the given concept-representation connection.

5. LIMITATIONS & FUTURE RESEARCH

Several limitations of the present analysis should be considered when interpreting these results. First, the study used a multiple-case design, which focuses on gaining in-depth insights into social processes that unfold over time rather than on generating generalizable evidence for causal effects. Therefore, this paper does not attempt to make causal claims about which prompts are effective, but to generate new hypotheses about social prompts. Based on

the theoretical consideration that instructional support for concept-representation connections may be most effective if it takes advantage of social prompts that different representation modes afford, one may hypothesize that instructional interventions should be designed to maximize instructors' capacity to assist students, regardless of the representation mode. One might also hypothesize that interventions with virtual representations are particularly effective if students are prompted (or trained) in monitoring their own understanding and communicate their meta-cognitive assessments to their partner. These hypotheses should be tested with study designs that allow for causal claims.

Another limitation regarding the generalizability stems from the focus on the representation dilemma; that is, how novice students see novel concepts in novel representations. Because students in this study had limited prior knowledge about concepts and representations, we do not know if the results generalize to advanced students. One may speculate that the importance of instructor support decreases as students learn, especially if students receive technology support. One might also speculate that the incremental way in which students focus on a concept or a representation alone before connecting them plays a lesser role if students have prior experience with representations or concepts. Hence, future research should examine social prompts among advanced students. A related limitation is that many utterances did not involve concept-representation connections. Consequently, the overall support and confidence for the discovered patterns is rather low. Concept-representation connections are one of many mechanisms of students' learning, so future research may apply the present analysis to other social (or conceptual) mechanisms of learning.

A further limitation results from this study's focus on social mechanisms that may underlie the complementary effects of representation modes on conceptual learning. Consequently, this study did not consider prompts beyond collaborative discourse, such as availability of resources in the classroom, an individual's bodily experiences with physical representations, etc. Future research could examine the role of such distributed and embodied types of prompts for concept-representation connections.

Finally, an assumption of this study was that concept-representation connections are a "desirable" educational outcome. While much research documents the importance of connecting concepts to representations for students' learning [1-12], this study did not test whether concept-representation connections correlate with learning outcomes. Future research could assess learning outcomes and test whether concept-representation connections mediate the effectiveness of physical and virtual representations and of interventions that combine both modes.

6. CONCLUSIONS

This study yields new theoretical insights into the representation dilemma by revealing how novice students connect new concepts to new representations. This study identified social events that prompt students to connect concepts to physical and virtual representations. These connections emerge in a co-constructive process that is incremental and requires instructor support. Meta-cognitive statements prompt students to help one another to make connections when an instructor is not always available.

At a practical level, this study yields new hypotheses suggesting that physical and virtual representations are most effective if instructor support is available. If instructor support is not available, interventions with virtual representations may benefit from meta-cognitive support. These hypotheses are empirically testable in studies on combinations of physical and virtual representations.

7. ACKNOWLEDGMENTS

We thank participating teachers and students, Sally Wu, Jamie Schuberth, Ashley Hong, Amber Kim, and Tae Ho Lee.

8. REFERENCES

[1] Dreher, A., and Kuntze, S.: 'Teachers facing the dilemma of multiple representations being aid and obstacle for learning: Evaluations of tasks and theme-specific noticing', Journal für Mathematik-Didaktik, 1-22 (2014)

[2] Uttal, D.H., and O'Doherty, K.: 'Comprehending and learning from 'visualizations': A developmental perspective', in Gilbert, J. (Ed.): 'Visualization: Theory and practice in science education' (Springer), 53-72 (2008)

[3] Gilbert, J.K.: 'Visualization: A metacognitive skill in science and science education', in Gilbert, J.K. (Ed.): 'Visualization: Theory and practice in science education' (Springer), 9-27 (2005)

[4] NRC: 'Learning to Think Spatially' (National Academies Press). (2006)

[5] Justi, R., and Gilbert, J.K.: 'Models and modelling in chemical education', in de Jong, O., Justi, R., Treagust, D.F., and van Driel, J.H. (Eds.): 'Chemical education: Towards research-based practice' (Kluwer Academic Publishers), 47-68 (2002)

[6] Ainsworth, S.: 'DeFT: A conceptual framework for considering learning with multiple representations.', Learning and Instruction, 16, 183-198 (2006)

[7] de Jong, T., Linn, M.C., and Zacharia, Z.C.: 'Physical and virtual laboratories in science and engineering education', Science, 340, 305-308 (2013)

[8] Zacharia, Z.C., Loizou, E., and Papaevripidou, M.: 'Is physicality an important aspect of learning through science experimentation among kindergarten students?', Early Childhood Research Quarterly, 27, 447-457 (2012)

[9] Olympiou, G., and Zacharia, Z.C.: 'Blending physical and virtual manipulatives: An effort to improve students' conceptual understanding through science laboratory experimentation', Science Education, 96, 21-47 (2012)

[10] Roschelle, J.: 'Learning by Collaborating: Convergent Conceptual Change', Journal of the Learning Sciences, 2, 235-276 (1992)

[11] Boulter, C.J., and Gilbert, J.K.: 'Challenges and opportunities of developing models in science education', in Gilbert, J.K., and Boulter, C.J. (Eds.): 'Developing Models in Science Education' (Kluwer Adademic Publishers), 343-362 (2000)

[12] Wu, H.K., Krajcik, J.S., and Soloway, E.: 'Promoting understanding of chemical representations: Students' use of a visualization tool in the classroom', Journal of research in science teaching, 38, 821-842 (2001)

[13] Donmoyer, R.: 'Generalizability and the single-case study', in Eisner, E., and Peshkin, A. (Eds.): 'Qualitative inquiry in education: The continuing debate' (Teachers College Press) 175-200 (1990)

[14] Rau, M.A.: 'Enhancing undergraduate chemistry learning by helping students make connections among multiple graphical representations', Chemistry Education Research and Practice, 16, 654-669 (2015)

[15] Romero, C., J.M. Luna, J.M., J.R. Romero, J.R., and S. Ventura, S.: 'RM-Tool: A framework for discovering and evaluating association rules', Advances in Engineering Software, 42, 566-576 (2011)

[16] Luna, J.M.: 'Pattern mining: Current status and emerging topics', Progress in Artificial Intelligence, 1-6 (2016)

Predicting Performance on MOOC Assessments using Multi-Regression Models

Zhiyun Ren
George Mason University
4400 University Dr,
Fairfax, VA 22030
zen4@masonlive.gmu.edu

Huzefa Rangwala
George Mason University
4400 University Dr,
Fairfax, VA 22030
rangwala@cs.gmu.edu

Aditya Johri
George Mason University
4400 University Dr,
Fairfax, VA 22030
johri@gmu.edu

ABSTRACT

The past few years has seen the rapid growth of data mining approaches for the analysis of data obtained from Massive Open Online Courses (MOOCs). The objectives of this study are to develop approaches to predict the scores a student may achieve on a given grade-related assessment based on information, considered as prior performance or prior activity in the course. We develop a personalized linear multiple regression (PLMR) model to predict the grade for a student, prior to attempting the assessment activity. The developed model is real-time and tracks the participation of a student within a MOOC (via click-stream server logs) and predicts the performance of a student on the next assessment within the course offering. We perform a comprehensive set of experiments on data obtained from two openEdX MOOCs via a Stanford University initiative. Our experimental results show the promise of the proposed approach in comparison to baseline approaches and also helps in identification of key features that are associated with the study habits and learning behaviors of students.

Keywords

Personalized Linear Multi-Regression Models, MOOC, Performance prediction

1. INTRODUCTION

Since their inception, Massive Open Online Courses (MOOCs) have aimed at delivering online learning on a wide variety of topics to a large number of participants across the world. Due to the low cost (most times zero) and lack of entry barriers (e.g., prerequisites or skill requirements) for the participants, large number of students enroll in MOOCs but only a small fraction of them keep themselves engaged in the learning materials and participate in the various activities associated with the course offering such as viewing the video lectures, studying the material, completing the various quizzes and homework-based assessments.

Given, this high attrition rate and potential of MOOCs to deliver low-cost but high quality education, several researchers have analyzed the server logs associated with these MOOCs to determine the factors associated with students dropping out. Several predictive methods have been developed to predict when a participant will drop out from a MOOC [4, 5, 6, 14].Using self reported surveys, studies have determined the different motivations for students enrolling and participating in a MOOC. Participants enroll in a MOOC sometimes to learn a subset of topics within the curriculum, sometimes to earn degree certificates for future career promotion or college credit, social experience or/and exploration of free online education [8]. Students with similar motivation have different learning outcomes from a MOOC based on the number of invested hours, prior education background, knowledge and skills [4].

In this paper, we present models to predict a student's future performance for a certain assessment activity witin a MOOC. Specifically, we develop an approach based on personalized linear multi-regression (PLMR) to predict the performance of a student as they attempt various graded activities (assessments) within the MOOC. This approach was previously studied within the context of predicting a student's performance based on graded activities within a traditional university course with data extracted from a learning management system (Moodle) [3]. The developed model is real-time and tracks the participation of a student within a MOOC (via click-stream server logs) and predicts the performance of a student on the next assessment within the course offering. Our approach also allows us to capture the varying studying patterns associated with different students, and responsible for their performance. We evaluate our predictive model on two MOOCs offered using the OpenEdX platform and made available for learning analytics research via the Center for Advanced Research through Online Learning at Stanford University [1].

We extract features that seek to identify the learning behavior and study habits for different students. These features capture the various interactions that show engagement, effort, learning and behavior for a given student participating in studying; by viewing the various video and text-based materials available within the MOOC offering coupled with student attempts on graded and non-graded activities like quizzes and homeworks. Our experimental evaluation shows accurate grade prediction for different types of homework as-

[1] datastage.stanford.edu

sessments in comparison to baseline models. Our approach also identifies the features found to be useful for predicting an accurate homework grade.

2. RELATED WORK

Several researchers have focused on the analysis of education data (including MOOCs), in an effort to understand the characteristics of student learning behaviors and motivation within this education model [11]. Brinton et. al. [1] developed an approach to predict if a student answers a question correct on the first attempt via click-stream information and social learning networks. Kennedy et. al. [7] analyzed the relationship between a student's prior knowledge on end-of-MOOC performance. Sunar et. al. [12] developed an approach to predict the possible interactions between peers participating in a MOOC. Elbadrawy et. al. [3] proposed the use of personalized linear multi-regression models to predict student performance in a traditional university by extracting data from course management systems (Moodle). Our study focuses on MOOCs, which presents different assumptions, challenges and features in comparison to a traditional university environment.

Most similar to our proposed work, Pardos et. al. proposed a model "Item Difficulty Effect Model" (IDEM) that incorporates the difficulty levels of different questions and modifies Bayesian Knowledge Tracing (BKT) model [2] by adding an "Item" node to every question node. By identifying the challenges associated with modeling MOOC data, the IDEM approach and extensions that involve splitting questions into several sub-parts and incorporating resource (knowledge) information [9] are considered state-of-the-art MOOC assessment prediction approaches and referred as KT-IDEM. However, this approach can only predict a binary value grade. In contrast, the model proposed in this paper is able to predict both, a continuous and a binary grade.

3. METHODS

3.1 Personal Linear Multi-Regression Models

We train a personalized linear multi-regression (PLMR) model [3] to predict student performance within a MOOC. Specifically, the grade $\hat{g}_{s,a}$ for a student s in an assessment activity a is predicted as follows:

$$\begin{aligned}\hat{g}_{s,a} &= b_s + p_s^t W f_{sa} \\ &= b_s + \sum_{d=1}^{l}(p_{s,d}\sum_{k=1}^{n_F} f_{sa,k} w_{d,k}),\end{aligned} \tag{1}$$

where b_s is bias term for student s, f_{sa} is the feature vector of an interaction between student s and activity a. The features extracted from the MOOC server logs are described in the next Section. n_F is the length of f_{sa}, indicating the dimension of our feature space. l is the number of linear regression models, W is the coefficient matrix of dimensions $l \times n_F$ that holds the coefficients of the l linear regression models, and p_s is a vector of length l that holds the memberships of student s within the l different regression models [3]. Using lasso [13], we solve the following optimization problem:

$$\underset{(W,P,B)}{minimize}\ L(W,P,B) + \gamma(\|P\|_F + \|W\|_F), \tag{2}$$

where W, P and B denote the feature weights, student memberships and bias terms, respectively. The loss function $L(\cdot)$ is the least square loss for regression problems. $\gamma(\|P\|_F + \|W\|_F)$ is a regularizer that controls the model complexity by controling the values of feature weights and student memberships. Tuning the scalar γ prevents model from over-fitting.

3.2 Feature Description

We extract features from MOOC server logs and formulate the PLMR model to predict real-time assessment grade for a given student. Figure 1 shows the various activities, generally available within a MOOC. Fig 1 (a) shows that each homework has corresponding quizzes, each of which has its corresponding video as resources for learning. Fig 1 (b) shows that while watching a video, a student can have a series of actions. Fig 1 (c) shows that while studying using a MOOC, a student can have several login sessions. In order to capture the latent information behind the click-stream for each student, we extract six types of features: (i) session features, (ii) quiz related features, (iii) video related features, (iv) homework related features, (v) time related features and (vi) interval-based features. These features constitute the feature vector f_{sa} for a student and a homework assessment. The description of these features are as follows:

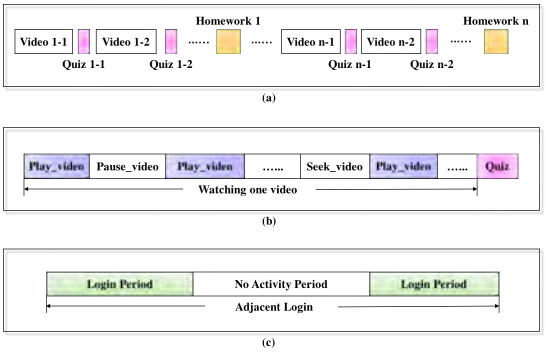


Figure 1: Different activities within a MOOC.

(i) Session features:.

A single study session is defined by a student login combined with the various available study interactions that a student may partake in. Since, students do not always log out of a session, we assume that a "no activity" period of more than one hour constitutes a student logging out of a session. We show a "no activity" period for a student between two consecutive sessions in Fig 1 (c).

- **NumSession** is the the average number of daily study sessions a student engages in, before a homework attempt.
- **AvgSessionLen** is the average length of each session in minutes. We calculate the average study time of a study session by

$$AvgSessionLen = \frac{Total\ study\ time}{NumSession}. \tag{3}$$

- **AvgNumLogin**. Students are free to choose when to login and study in a MOOC environment. We consider

a day as a "work day" if a student logs into the study system; and a day as "rest day" if a student does not. The rate of "work" and "rest" can capture a student's learning habits and engagement characteristics.

$$AvgNumLogin = \frac{\#\ of\ \text{“}work\ day\text{”}}{\#\ of\ \text{“}work\ day\text{”}\ +\#\ of\ \text{“}rest\ day\text{”}}. \quad (4)$$

(ii) Quiz Related features:

- **NumQuiz** is the number of quizzes a student takes before a homework attempt. This feature reflects the student's dedication towards the course material and a factor towards performance in a homework.
- **AvgQuiz** is the average number of attempts for each quiz. The MOOCs studied in this paper allow unlimited attempts on a quiz.

(iii) Video Related features:

- **VideoNum** denotes the number of distinct video sessions for a student before a homework attempt.
- **VideoNumPause** is the average number of pause actions per video. There are several actions associated with viewing videos, including "pause video", "play video", "seek video" and "load video". Tracking these actions allows for capturing a student's focus level and learning habits.
- **VideoViewTime** is the total video viewing time.
- **VideoPctWatch**. In a large amount of cases, students do not finish watching a full video. As such, we calculate the average percentage of the watched part of a video.

(iv) Homework Related features:

- **HWProblemSave** is the average number of "save answer" actions for each homework assessment. Before submitting answers for a homework, students are allowed to save their answer sheet and check as many times as they need. This feature is more valuable when the MOOC provides only one chance for a homework answer submission.

(v) Time Related features:

- **TimeHwQuiz** is the time between a homework answer submission and the last quiz attempt.
- **TimeHwVideo** is the time between a homework answer submission and the last video watching activity.
- **TimePlayVideo** is the percentage of study sessions with video watching activity over all the study sessions.
- **HwSessions** is the number of sessions that have homework related activities (save and submit).

(vi) Interval-Based features:.
It is expected that there will be some changes in study activities once the students know the former homework's grade. They may study harder if they don't get a satisfactory score. The interval-based features are aiming to represent different activities between two consecutive homeworks.

- **IntervalNumQuiz**: denotes the number of quizzes the student takes between two homeworks.
- **IntervalQuizAttempt**: is the average number of quiz attempts between two homeworks.
- **IntervalVideo**: is the number of videos a student watches between two homeworks.
- **IntervalDailySession**: is the average number of sessions per day between two homeworks.
- **IntervalLogin**: is the percentage of login days between two homeworks.

We also use the cumulative grade (so-far) on quizzes and homeworks for a student as a feature and denote it by **Meanscore**. For our baseline approach we only consider the averages computed on the previous homeworks.

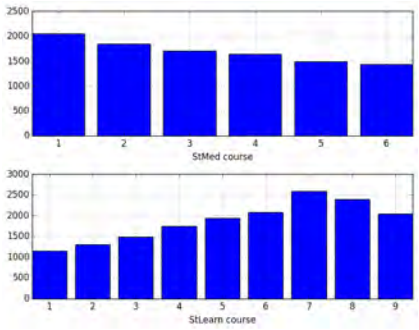


Figure 2: Distribution of students attempting each Assessment. StMed and StLearn had 6 and 9 assessments, respectively.

4. EXPERIMENTS

4.1 Datasets

We evaluated our methods on two MOOCs: "Statistics in Medicine" (represented as StMed in this paper) taught in Summer 2014 and "Statistical Learning" (represented as StLearn in this paper) taught in Winter 2015.

StMed: This dataset includes server logs tracking information about a student viewing video lectures, checking text/web articles, attempting quizzes and homeworks (which are graded). Specifically, this MOOC contains 9 learning units with 111 assessments, including 79 quizzes, 6 homeworks and 26 single questions. The course had 13,130 students enrolled, among which 4337 students submitted at least one assignment (quiz or homework) and had corresponding scores, 1262 students have completed part of the six homeworks and 1099 students have attempted all the homeworks. 193 students attempted all the 79 quizzes and six homeworks. This course had 131 videos and 6481 students had video related activity.

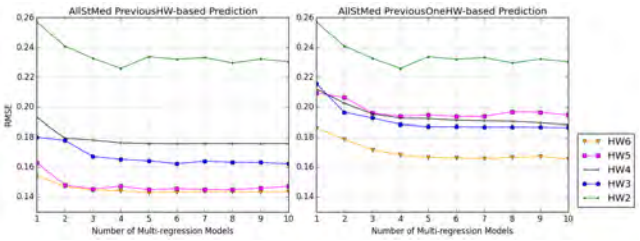


Figure 3: AllStMed Prediction Results. RMSE (↓ is better).

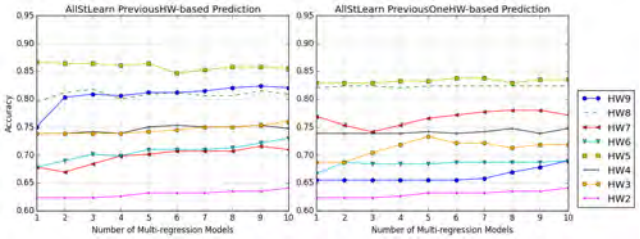


Figure 4: AllStLearn Prediction Results. Accuracy (↑ is better).

StLearn: This course had ten units. Except the first one, all units have quizzes and end of unit homeworks, which add up to 103 assessments in total. 52,821 students enrolled in this course, and 4987 students had assessment activities, 3509 students attempted a subsets of the available homeworks while 346 students attempted all the 9 homeworks, and 118 students attempted all the 103 assessments. The key difference between the homeworks in the StLearn in comparison to the StMed is that homeworks have only one question which a student can either get correct or incorrect. As such, scoring in this MOOC is binary instead of continuous. To predict whether a student answers a question correctly, we reformulate the regression problem as a classification problem using a logistic loss function. Figure 2 shows the distribution of students attempting the different assessments available across the two MOOCs studied here.

4.2 Experimental Protocol

In order to gain a deep insight of students' performance in a MOOC, we perform two types of experiments. Given n, homework assessments represented as $\{H_1, \ldots, H_n\}$ our objective is to predict the score a student achieves in each of the n homeworks. Depicting the most realistic setting, for the i-th homework, H_i we define the training set as all homework and student pairs who attempt and have a score for all homeworks up to the H_{i-1}. For predicting the score for H_i for a given student, we use all the features extracted just before attempting the target homework H_i. We refer to this as **PreviousHW-based Prediction**. Secondly, for the predicting i-th homework H_i's score, we use training data of student-homework pairs restricted from only the previous one homework i.e., H_{i-1}. This experiment is referred by **PreviousOneHW-based Prediction**. Note, in these cases we cannot make any prediction for the first homework (H_1) since, we do not have any training information for a given student.

4.3 Data Partition

We partition the students for StLearn and StMed into two groups: the group of students who attempt *all* the requested homeworks, and the group of students who finish *few* of the homeworks. This allows us to consider the different motivations and expectations of students enrolling in a MOOC. For example, the students who aim to learn in a MOOC may choose watching videos over taking all homeworks. While, the students who want to achieve a degree certificate may focus on the homework completeness. We refer to the first group by "Partial homeworks accomplished group", and the second group by "All homeworks accomplished group". We evaluate our models on the two groups for the **AllStMed** and **AllStLearn** datasets. Specifically, we name the four group of students as **AllStMed**, **AllStLearn**, **PartialStMed** and **PartialStLearn** based on their group and MOOC class.

HW#	PLMR	Meanscore
2	**0.230**	0.248
3	**0.162**	0.176
4	**0.176**	0.196
5	**0.144**	0.156
6	**0.143**	0.150
Avg	**0.171**	0.185

Table 1: PreviousHW-based RMSE Performance (RMSE) comparison for AllStMed.

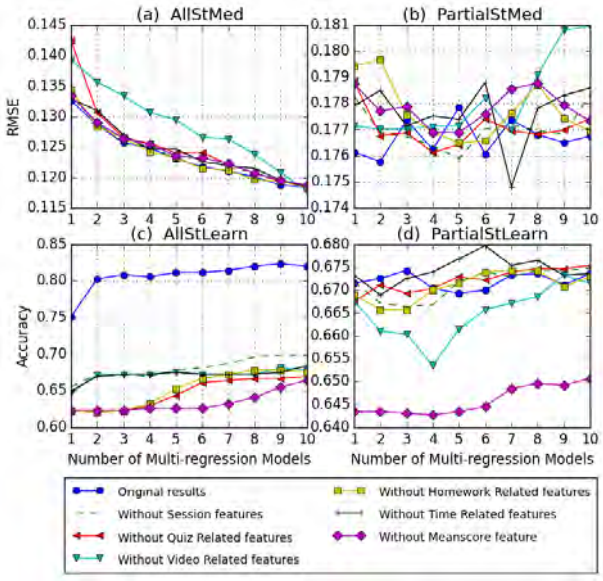


Figure 5: Predictive Performance with Removal of Feature Types.

4.4 Evaluation Metrics

StMed course has continuous scores for a homework, which are scaled between 0 and 1. However, the homework score is binary in the StLearn course, indicating whether the student answers a question correctly or incorrectly. For StLearn, we use a logistic loss and formulate a classification problem

HW#	Accuracy (↑)			F_1 (↑)		
	PLMR	Baseline		PLMR	Baseline	
		Meanscore	KT-IDEM		Meanscore	KT-IDEM
2	0.641	**0.646**	0.623	0.775	**0.777**	0.768
3	**0.760**	0.580	0.681	**0.821**	0.805	0.810
4	**0.754**	0.710	0.739	0.838	0.706	**0.850**
5	**0.867**	0.809	0.829	**0.920**	0.880	0.906
6	**0.730**	0.678	0.667	**0.808**	0.776	0.800
7	0.716	0.675	**0.730**	**0.887**	0.878	0.844
8	**0.817**	0.762	0.817	**0.903**	0.849	0.886
9	**0.823**	0.794	0.777	**0.864**	0.856	0.853
Avg	**0.764**	0.707	0.759	**0.852**	0.816	0.848

Table 2: PreviousHW-based prediction performance comparison for AllStLearn group.

instead of the regression problem as done for the StMed course. To evaluate the performance of our approach, we use the root mean squared error (RMSE) as the metric of choice for regression problem. For classification problem, we use accuracy and the F1-score (harmonic mean of precision and recall), known to be a suitable metric for imbalanced datasets.

4.5 Comparative Approaches.

In this work, we compare the performance of our proposed methods with two different competitive baseline approaches.

(i) **Average grade of the previous homeworks.** We calculate the mean score of a given student's previous homeworks to predict their future performance and is denoted as Meanscore. We use this method to compare our prediction results on StMed.

(ii) **KT-IDEM [10].** KT-IDEM is a modified version of original BKT model. By adding an "item" node to every question node, the model is able to identify different difficulty levels of each question. Since this model can only predict a binary value grade, we use this model to compare our prediction results on StLearn.

5. RESULTS AND DISCUSSION

5.1 Assessment Prediction Results

Figures 3 and 4 show the prediction results with varying number of regression models for the AllStMed and AllStLearn MOOCs, respectively. Analyzing Figure 3 we observe that as the number of regression models increases the RMSE metric goes lower and use of five models seems to be good choice for all the different homeworks. Comparing the PreviousHW- and PreviousOneHW-based results, we notice that predictions for all the homeworks (HW3, HW4, HW5, and HW6) benefits from using all the available training data prior to those homeworks i.e., to predict grade for H_i it is better to use training information extracted from $H_1 \dots H_{i-1}$ rather than just H_{i-1}. Similar observations can be made while analyzing the prediction results for the AllStLearn cohort which includes nine homework correct/incorrect binary assessments. Figure 4 shows the accuracy scores (higher is better) for the three experiments. For the PreviousOneHW- and PreviousHW-based experiments HW5 shows the best prediction results. This suggests that in the middle of a MOOC, students tend to have stable study activities and the performance is more predictable than other phases. Also, some homeworks thrive well with just using training data from the previous homework (PreviousOneHW-based, e.g. HW3).

5.1.1 Comparative Performance

Table 1 shows the comparison between baseline approach (Meanscore) and the predictive model for the PreviousHW-based experiments for the AllStMed group. We cannot report results for the KT-IDEM model since, it solves the binary classification problem only. Table 2 shows the comparison of the accuracy and F1 scores of the AllStLearn groups with baseline approaches. We notice that for predicting the second homework, which only uses the information from HW1, the predictive model is not as good as the mean baseline, which reflects that under the situation of lack of necessary amount of information, linear regression models cannot always outperform the baseline. But as the dataset gets larger, our approach outperforms the baseline due to the availability of more training data. From Table 2, we also notice for some homework, KT-IDEM has better performance than PLMR (HW7 and HW4). This could be due to unstable academic activities during these two study periods, which can effect the performance of PLMR.

5.1.2 Feature Importance

We test the effect of each feature set in predicting the assessment scores by training the models under the absence of each feature group. For the StLearn course, since there is no limit on homework attempts, we do not add Interval-Based feature groups to the predictive model. Figure 5 shows the comparison of each prediction result for AllStMed, PartialStMed, AllStLearn and PartialStLearn cohorts. Analyzing these results we observe that for the StLearn MOOC, meanscore is a significant feature and removing it leads to a substantial decrease in accuracy for both All and Partial-cohorts. For the AllStMed, the removal of video related features leads to the most decrease in performance (i.e., increased RMSE). This suggests that features related to the video watching are crucial for predicting the final homework scores. For the PartialStMed, the use of all feature types or a subset does not show a clear winner. This could be due to the varying characteristics of students within these group.

Another way to analyze feature importance is to exclude the influence of the dominant feature, which is meanscore in our study. The evaluation formula of the importance of the i_{th} feature (excluding meanscore feature) is as follows:

$$I_i = \frac{1}{N}\sum_{n=1}^{N}\frac{\sum_{d=1}^{l}|p_{n_S,d}f_{n_S,i}w_{d,i}|}{\sum_{d=1}^{l}|p_{n_S,d}\sum_{k=1}^{n_F}f_{n_S,k}w_{d,k}|}, \qquad (5)$$

where N is number of test samples, n_S is the student number corresponding to the n_{th} test sample. $f_{n_S,i}$ is the feature value of an interaction between student n_S and activity i. n_F is the number of features. l is the number of linear regression models. $w_{d,i}$ is the coefficient of d_{th} linear regression model with i_{th} feature, and $p_{n_S,d}$ is the membership of student n_S with the d_{th} regression model. We calculate each feature's importance by calculating the percentage contribution of each feature to the overall grade prediction. Figure 6 shows the feature importance on the AllStMed group, excluding Meanscore feature. We can see **NumQuiz** and **VideoPctWatch** are the most important for AllStMed group besides **Meanscore** feature.

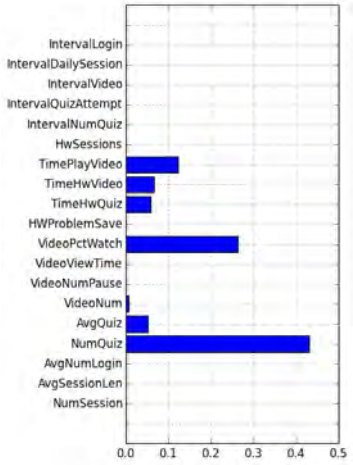


Figure 6: Feature importance for AllStMed.

6. CONCLUSION AND FUTURE WORK

In this work we formulated a personalized multiple linear regression model to predict the homework grades for a student enrolled and participating within a MOOC. Our contributions include engineering features that capture a student's studying behavior and learning habits, derived solely from the server logs of MOOCs. We evaluated our framework on two OpenEdX MOOC courses provided by an initiative at Stanford University. Our experimental evaluation shows improved performance in terms of prediction of real time homework scores compared to baseline methods. We also studied on different groups of student participants due to their motivation. Features associated with engagement (logging multiple times), studying materials (viewing videos and attempting quizzes) were found to be important along with prior homework scores for this prediction problem.

7. ACKNOWLEDGEMENTS

Funding was provided by NSF Grant, 1447489.

8. REFERENCES

[1] Christopher G Brinton and Mung Chiang. Mooc performance prediction via clickstream data and social learning networks. *To appear, 34th IEEE INFOCOM. IEEE*, 2015.

[2] Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[3] Asmaa Elbadrawy, Scott Studham, and George Karypis. Personalized multi-regression models for predicting students performance in course activities. *UMN CS 14-011*, 2014.

[4] Jeffrey A Greene, Christopher A Oswald, and Jeffrey Pomerantz. Predictors of retention and achievement in a massive open online course. *American Educational Research Journal*, page 0002831215584621, 2015.

[5] Glyn Hughes and Chelsea Dobbins. The utilization of data analysis techniques in predicting student performance in massive open online courses (moocs). *Research and Practice in Technology Enhanced Learning*, 10(1):1–18, 2015.

[6] Suhang Jiang, Adrienne Williams, Katerina Schenke, Mark Warschauer, and Diane O'dowd. Predicting mooc performance with week 1 behavior. In *Educational Data Mining 2014*, 2014.

[7] Gregor Kennedy, Carleton Coffrin, Paula de Barba, and Linda Corrin. Predicting success: how learners' prior knowledge, skills and activities predict mooc performance. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, pages 136–140. ACM, 2015.

[8] Daniel FO Onah and Jane Sinclair. Learners expectations and motivations using content analysis in a mooc. In *EdMedia 2015-World Conference on Educational Media and Technology*, volume 2015, pages 185–194. Association for the Advancement of Computing in Education (AACE), 2015.

[9] Zachary Pardos, Yoav Bergner, Daniel Seaton, and David Pritchard. Adapting bayesian knowledge tracing to a massive open online course in edx. In *Educational Data Mining 2013*, 2013.

[10] Zachary A Pardos and Neil T Heffernan. Kt-idem: Introducing item difficulty to the knowledge tracing model. In *User Modeling, Adaption and Personalization*, pages 243–254. Springer, 2011.

[11] Alejandro Peña-Ayala. Educational data mining: A survey and a data mining-based analysis of recent works. *Expert systems with applications*, 41(4):1432–1462, 2014.

[12] Ayse Saliha Sunar, Nor Aniza Abdullah, Susan White, and Hugh C Davis. Analysing and predicting recurrent interactions among learners during online discussions in a mooc. *Proceedings of the 11th International Conference on Knowledge Management*, 2015.

[13] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[14] Jacob Whitehill, Joseph Jay Williams, Glenn Lopez, Cody Austun Coleman, and Justin Reich. Beyond prediction: First steps toward automatic intervention in mooc student stopout. *Available at SSRN 2611750*, 2015.

Validating Game-based Measures of Implicit Science Learning

Elizabeth Rowe[1]
Jodi Asbell-Clarke[2]
Teon Edwards[7]
EdGE @ TERC
2067 Massachusetts Ave
Cambridge, MA 02140
elizabeth_rowe@terc.edu
Jodi_asbell-clarke@terc.edu
teon_edwards@terc.edu

Michael Eagle[3]
Carnegie Mellon University
4902 Forbes Ave
Pittsburgh, PA 15213
meagle@cs.cmu.edu

Drew Hicks[4]
Tiffany Barnes[5]
Rebecca Brown[6]
NC State University
890 Oval Drive
Raleigh, NC 27606
drew@drewhicks.com
tmbarnes@ncsu.edu
rabrown@ncsu.edu

ABSTRACT
Building on prior work visualizing player behavior using interaction networks [1], we examined whether measures of implicit science learning collected during gameplay were significantly related to changes in external pre-post assessments of the same constructs. As part of a national implementation study, we collected data from 329 high school students playing an optics puzzle game, *Quantum Spectre*, and modeled their gameplay as an interaction network, examining errors hypothesized to be related to a lack of implicit understanding of the science concepts embedded in the game. Hierarchical linear modeling (HLM) showed a negative relationship between the science errors identified during gameplay and implicit science learning. These results suggest *Quantum Spectre* gameplay behaviors are valid assessments of implicit science learning. Implications for how gameplay data might inform classroom teaching in-game scaffolding is discussed.

Keywords
Game-based learning, Interaction Networks, Implicit Science Learning, Hierarchical linear modeling

1. INTRODUCTION
As digital games become increasingly prevalent in today's society and are played by the majority of youth of all demographics [2], it behooves us to study how the energy and passion invested in gaming can be harnessed for productive purposes. Game-based learning interests education researchers and learning scientists because digital games uniquely engage learners and because their data logs can serve as input for innovative learning assessments [3]. Data logs generated through gameplay can be used to study players' in-game activity [4] and how game-based learning can be leveraged for classroom learning. Research shows that elements of gameplay can invoke complex thinking such as scientific inquiry [5] and may foster learning-related skills such as creativity and persistence [4].

This work examines complex behaviors of students solving optics puzzles in the educational game *Quantum Spectre,* using interaction networks. An *Interaction Network* is a complex network representation of all observed player-game interactions for a given problem or task in a game or tutoring system [6]. Regions of the network can be discovered by applying network clustering methods. These regions correspond to high-level student approaches to problems [7]. In this work, we used Interaction Networks as visualizations to analyze *Quantum Spectre* gameplay data and automated the coding of game states that correspond to incorrect applications of the game's core science concepts. Three types of errors were coded: two science errors (placement and rotation) and puzzle errors.

This paper reports HLM analyses that relate those coded game states to implicit science learning measured by external pre/post assessments. The analyses examine how game-based learning is a function of *what players do* in the game, not simply duration of gameplay or highest level reached. This information is useful for building an adaptive version of the game to scaffold players' implicit science learning and for informing teachers about important aspects of student competency.

2. IMPLICIT SCIENCE LEARNING
Polanyi argued that implicit knowledge (also called tacit knowledge) is foundational and a required element of explicit learning [8]. Implicit understandings are embodied and enacted through our interactions with the world around us, but may not yet be formalized or expressed verbally or textually. Vygotsky described similar abilities and understandings a learner brings to a learning situation that can be scaffolded by a teacher, environment, and tools [9]. Implicit misunderstandings (often called misconceptions) may get in the way of a learner's conceptual development [10, 11], particularly in the area of basic physics, such as Newton's Laws of Motion. The work of diSessa distinguishes between the intuitive knowledge that novices hold—a book will not fall through a table or a glowing filament is hot—from an expert understanding of these phenomena, explaining that while learners' behaviors may be guided by implicit understandings, the learner is not necessarily ready to express the related formalisms or question the ideas in a deeper sense [12].

Games promise to reveal implicit learning because they can be (a) "sticky"—meaning they encourage players to dwell in the phenomena and (b) they leave a digital trail that reveals the patterns the players used in their learning process. Several

researchers have used educational data mining techniques within an Evidence-Centered Design framework to develop stealth assessments that discern evidence of learning from the vast amount of click data generated by online science games such as *SimCityEDU* [13], *Physics Playground* [14], and *Surge* [15].

As players "level up" in a game, they typically deal with the mechanics in increasingly complex applications, building implicit knowledge about the underlying system. Because games allow players to fail, repeat, revise, and try again—recording what players do in the process—games may be powerful formative assessments of learning, and the strategies players build. The methods players use to tackle new challenges may demonstrate conceptual understanding that the learner may not express in other ways and that may not be measured by current external learning assessments [4, 16]. Careful alignment of game mechanics with learning and assessment mechanics [17] may reveal implicit learning and empower teachers and learners to help bridge game-based knowledge to other forms of learning.

In a classroom, teachers may be able to build on implicit game-based learning if they have the right information and tools to support students at key moments in the learning process. That may consist of real-time information, provided during class to know who is struggling and needs attention, or more reflective information after school to help plan lessons for the next day based on class gameplay [18]. Post-game debriefing and discussions connecting gameplay with classroom learning help students apply and transfer learning that takes place in games [19]. To exploit learning that happens in games, teachers need to build bridges between the students' "aha" moments while playing [20] and the content being covered in the classroom.

3. QUANTUM SPECTRE

To examine implicit science game-based learning, we studied high school students playing a Physics-oriented game called *Quantum Spectre. Quantum Spectre* is a puzzle-style game, designed for play in browsers and on tablets (Figure 1).

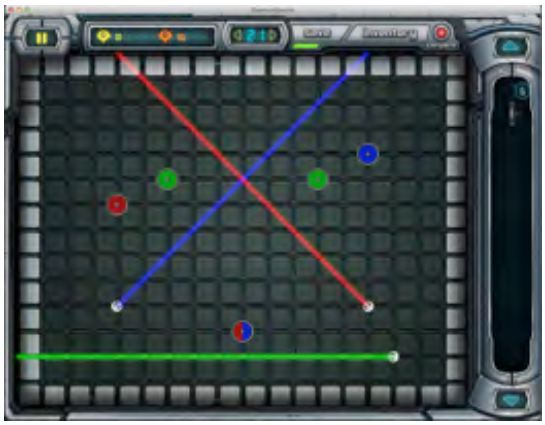


Figure 1: *Quantum Spectre* Puzzle 21. Players must direct the laser beams to the matching colored targets using movable mirrors and other optical devices, selected from the inventory on the right.

Players use optical devices, such as lenses and mirrors, to guide colored laser beams to matching targets. The lenses and mirrors can be flat, convex, or concave and single or double-sided. All devices produce scientifically accurate results when interacting with the laser beams. When the laser beams in a puzzle reach the matching colored targets, the puzzle is solved (i.e., goal state is reached) and the player is scored on the number of moves used. The player earns three stars if the puzzle has been solved in the optimal number of moves, two stars for a low number of extra moves, and one star for simply solving the puzzle. Regardless of their score, players can proceed onto the next level, but players can repeat earlier levels at any time to improve their performance.

The game is divided into 6 zones with 30 puzzles in each zone. In Zone 1 of *Quantum Spectre*, the puzzles focus on 2 key concepts:

- The **Law of Reflection**, or Angle of Incidence equals Angle of Reflection—When reflecting off of a smooth surface, the path of a ray of light (such as a laser beam) will make the same angle with the surface (relative to the normal) upon exit as it makes upon entry.
- **Slope**—Players can use the squares on the game grid and calculate the slope (rise over run) to figure out and/or predict the paths of laser beams and where to place items.

This study focuses on data from Puzzles 14-23 in Zone 1 of the game. At this point in gameplay, players have presumably mastered the game mechanic, and mastery of the puzzles typically requires an understanding of Slope and the Law of Reflection. Table 1 provides an overview of Puzzles 14-23. The number of goal states reflects the number of unique solutions (position-rotation combinations) for each puzzle.

Table 1: *Quantum Spectre* Puzzles 14-23

Game Level	# Mirrors	# Targets	# Optimal Moves	# Goal States
14	1	1	2	1
15	2	1	4	5
16	2	1	3	8
17	2	2	4	1
18	2	2	4	6
19	4	4	7	4
20	6	3	12	42
21	6	5	11	6
22	3	1	6	1
23	4	2	8	3

4. CLASSIFYING GAMEPLAY BEHAVIORS USING INTERACTION NETWORKS

To simplify the vast number of puzzle solution paths into a manageable group we could study, we used a method called Interaction Networks (INs). INs use a complex network data structure to represent players' solutions as traces of game states and actions, with additional information such as edge labels (e.g., labels of player actions). This process involved 4 key steps [1]: creating a full IN for each puzzle, clustering player actions using laser shapes, classifying clusters for evidence of implicit science understanding, and automating coding of player actions.

4.1 Create Full Interaction Network

To construct an IN, we collected the set of all solution attempts for that puzzle. Each interaction is defined as Initial State, Action, and Resulting State, from the start of the puzzle until the player

solves the puzzle or exits the system. A sample trace is shown in Figure 2. Player actions are represented as edges in the network.

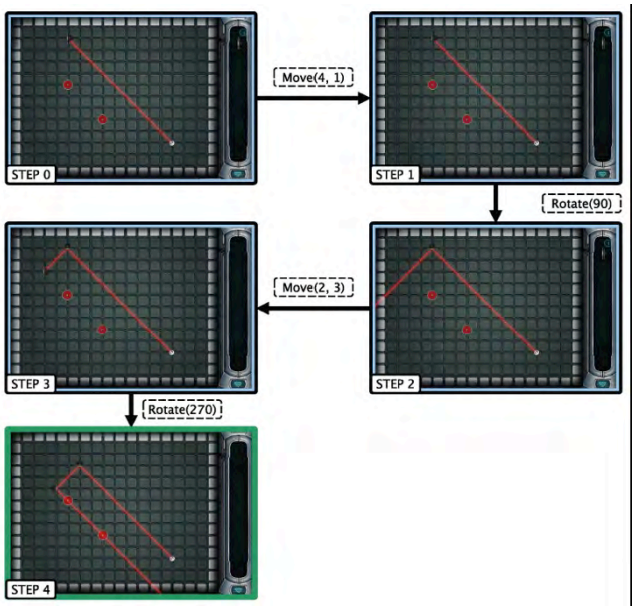


Figure 2: Sample trace of player actions in *Quantum Spectre* Puzzle 18 of Zone 1

Table 2 describes the complexity of the full interaction networks for Puzzles 14-23 for the full sample of students playing the game. The full IN of every state and every action taken was large, complex, and difficult to interpret in terms of player understanding.

Table 2: Interaction Networks in Puzzles 14-23

Game Level	# Players	Total # Moves	# Network Edges	# Unique States	# Laser Shapes
14	479	3003	462	164	5
15	473	3866	1009	484	10
16	462	3218	761	446	12
17	454	10878	1899	1067	21
18	439	10314	3458	1800	22
19	416	15389	7093	4550	330
20	384	10778	4947	2391	264
21	349	23080	13919	6261	696
22	282	3697	1500	1017	146
23	271	10529	6154	4138	364

4.2 Cluster States by Laser Shapes

Most puzzles have states in which different configurations of objects result in similar output. These states could be considered equivalent since they show the same player proficiencies or errors, but a simple state representation would consider them as different states. In previous work using INs for games, it has been helpful to consider the output of a state as well as the position/orientation of objects in that state [7]. To group these equivalent states, we took a similar approach, using "laser shape" as part of our state representation to create Approach Maps. Approach Maps are a visual summary of the information contained in the interaction network [7]. This reduction is created by grouping similar states together based on how often students co-visit the states during their solution attempts. Here, the approach map consists of a list of targets hit by a laser of the appropriate color and a list of angles taken by that laser. This allows game states that represent similar errors to be effectively grouped together, as shown in Figure 3.

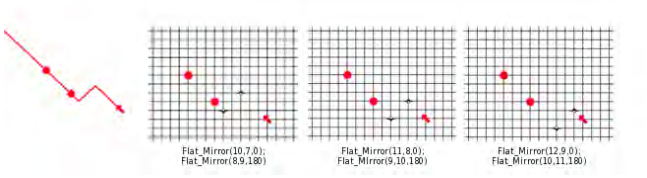


Figure 3: Using laser shape to group similar game states in Puzzle 18.

This approach preserves the relevant properties of a board state while ignoring distance traveled, which is not relevant to the game state.

4.3 Classify Player Actions for Implicit Science Understanding

A *Quantum Spectre* game designer who has a science education background, worked with a researcher to classify each laser shape into one of three categories:

1) *Correct move*—placement and rotation of the mirror are consistent with an eventual goal state
2) *Placement errors*—placement of the mirror in a location that does not match a goal state—may indicate a lack of understanding of slope.
3) *Rotation errors*—rotation of a mirror to an angle that does not match a goal state—may indicate a lack of understanding of the Law of Reflection.

As described elsewhere [1] using a subset of these data, the game designer and researcher also identified placements that were not consistent with a goal state but were more indicative of a lack of grasp of the puzzle mechanic than of a lack of science understanding. We labeled these *Puzzle errors.* For example, in puzzle shown in Figure 2, a correct solution requires players to use the two available mirrors to direct the laser through the two targets simultaneously. In Figure 4, player actions are consistent with someone who understands slope (i.e., they placed the mirror on the path of the laser) and the Law of Reflection (i.e., they rotated the mirror to reflect the mirror through the target). However, their actions are not going to let them solve this puzzle.

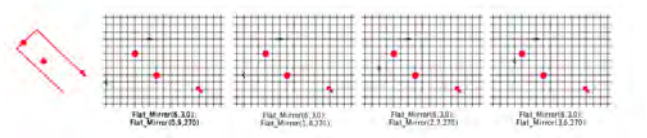


Figure 4: Sample Puzzle Errors in Puzzle 18.

4.4 Automated Coding of Individual Player Behaviors

Once all laser shapes had been coded and puzzle error placements identified, we automated the coding of individual player behaviors. Every player behavior was classified as a Placement Error, Rotation Error, or Puzzle Error (0=Not Present; 1=Present). These are mutually exclusive player behaviors. Player actions with none of these errors were classified as Correct. Figure 5 shows the distribution of player behaviors across each puzzle.

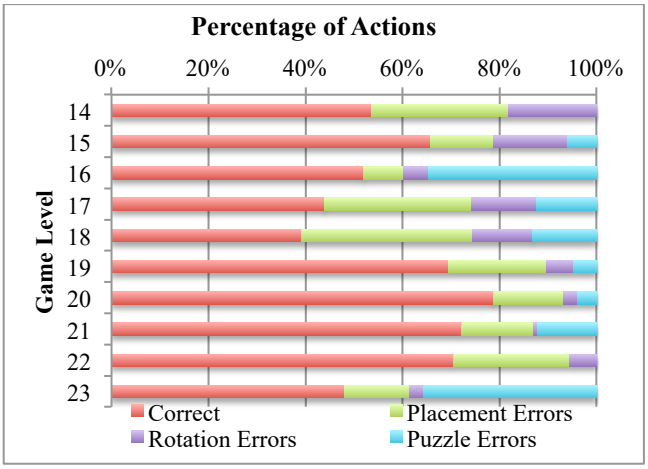


Figure 5: Error rates by puzzle level

The percentage of correct moves ranged from 39% in Level 18 to 79% in Level 20. Placement error rates range from 8% (Level 16) to 35% (Level 18). Rotation error rates were most common in earlier puzzles, 35% in Level 14 to 1% in Level 21. In two puzzles, Levels 14 and 22, no puzzle errors were possible. Puzzle errors in the remaining puzzles ranged from 4% (Level 20) to 36% (Level 23).

5. RESEARCH QUESTIONS & HYPOTHESES

In this paper, we examine the ways in which the extent of players' puzzle and science errors are related to changes in their performance on a pre-post assessment of slope and the Law of Reflection. We anticipated a negative relationship between placement errors, rotation errors, and pre-post assessment results—that is players who are demonstrating a lack of understanding of the science concepts in their gameplay will have smaller gains than players whose gameplay is consistent with an implicit understanding of slope and the Law of Reflection. Our anticipated relationship between puzzle errors and pre-post assessment results was less clear. It could be that puzzle errors interfere with their implicit learning of the science content. It could also be players who understand the science content are just as likely to make puzzle errors as players without that understanding, so there may be no relationship between the number of puzzle errors and pre-post assessment results.

6. METHODS

Teachers were assigned to one of three groups as part of a national *Quantum Spectre* implementation study. In Bridge classrooms, teachers encouraged students to play the game outside of class and used examples from the game as part of their science instruction. In Game Only classrooms, teachers encouraged students to play the game but provide no game examples during their science instruction. In Control classrooms, teachers and students did their normal science instruction with their students not knowing about the game. This paper reports gameplay data from the 329 students in 29 classes (14 Bridge and 15 Game Only) that participated in the implementation study during the 2013-14 and 2014-15 academic years.

6.1 Sample

Because this study focuses on Puzzles 14-23 in Zone 1 of the game, 79 students were excluded from these analyses because they did not attempt Puzzle 14 of the game. The final sample of 329 high school science students included 132 females, 162 students in Bridge classrooms, 281 students in non-Honors/AP classrooms, and 249 students in classrooms where more than 75 percent of the students participated in the study.

6.2 Measures

This study collected gameplay log data, as described above, as well as pre-post assessment and student/classroom characteristics,.

6.2.1 Gameplay Metrics

To allow for the fact that students (a) used varying numbers of moves to solve the puzzles and (b) not all students completed Levels 14-23; the percentage of the total number of moves (actions) that were correct, placement errors, rotation errors, and puzzle errors was calculated. The mean error rate across all students was 19% placement errors, 7% rotation errors, and 12% puzzle errors. We used standardized (z-scores) error rates.

The total amount of time each student played *Quantum Spectre* and the highest level reached were also recorded. Previous analyses showed Puzzle 21 to have a high dropout rate [21], we analyzed whether or not players completing Puzzle 21 had any relationship to changes in pre-post assessment results. Among this sample, there was no significant difference in the percentage of students in Bridge and Game Only classrooms that reached Puzzle 22 (X^2=3.53, 1 d.f., p=0.06). Given the non-normal distribution of the amount of time students played *Quantum Spectre*, we categorized students as having played less than 1 hour, or 1 hour or more. Forty-one percent of students played 1 hour or more, this proportion did not vary among students in Bridge and Game Only classrooms (X^2=3.23, 1.d.f., p=0.07).

6.2.2 Students & Classroom Characteristics

When completing the pre-assessment, students were asked to indicate their gender. We categorized class names (e.g., Honors Physics 101) obtained from teacher applications as being either Honors/AP classes or not. Seven of the 29 classes in this study were Honors/AP classes. Finally, we asked teachers the total number of students enrolled in each class. We calculated the percentage of the class with complete study information (e.g., complete consent/assent forms, pre-post assessments complete, and gameplay beyond Puzzle 1 in Zone 1). This ranged from 31 to 100 percent of each class, with the majority of classes (26) having more than half of the students participating.

6.2.3 Assessments

Science content experts developed assessment instruments and tested them in a series of think-aloud interviews with 10 high school students. Each assessment contained 12 (pre) and 13 (post) questions that required minimal formalisms to complete. The pre- and post-assessments each included 3 items related to focal length that are not included in these analyses. Figures 6 and 7 are sample items for slope and the Law of Reflection, respectively. In Figure 6, students are asked which point (A-D) a line drawn through the two black points would hit. The item in Figure 7 asks students which letter each laser would hit.

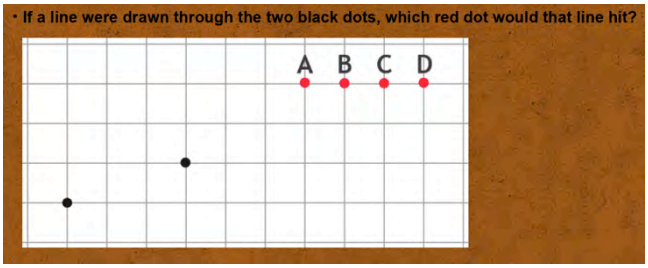


Figure 6: Sample Slope assessment item

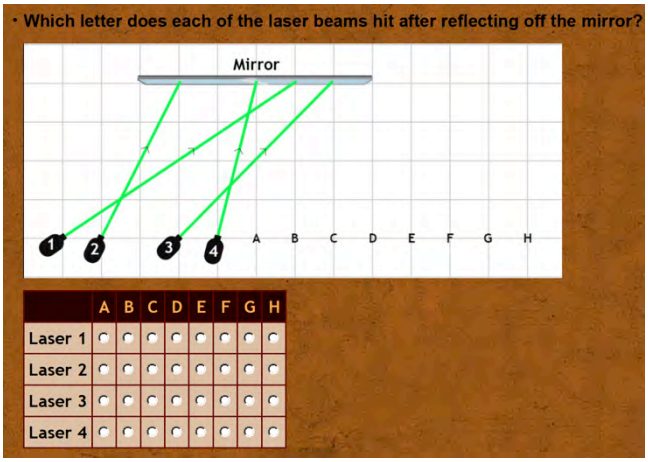


Figure 7: Sample Law of Reflection assessment item

These analyses are limited to the 9 pre and 10 post items focused on slope and the Law of Reflection. These pre- and post-assessment items had good internal consistency (Cronbach's alpha was 0.70 (pre) and 0.73 (post). To account for the different number of items, we used the percentage of items answered correctly in the analyses. Students answered an average of 53 percent of the pre assessment items and 59 percent of the post assessment items correctly. Students in Bridge classrooms, however, answered significantly fewer questions correctly on both the pre- and post-assessment than students in Game Only classrooms ($F=19.2$, 1, 132 d.f., $p<0.01$). On average, students in Bridge classrooms answered 48 percent of the pre-assessment and 55 percent of the post-assessment items. In contract, students in Game Only classrooms answered 58 percent of the pre assessment items and 63 percent of the post assessment items correctly.

7. RESULTS

Using the SPSS MIXED linear models procedure, HLM analyses began with an unconditional 3-level model with students, classrooms, and teachers using Restricted Maximum Likelihood (REML) and unstructured covariances. In the 3-level model, seven percent of the variation was at the teacher level. Triple that proportion of the overall variation was attributable to the classroom level. A 2-level unconditional model with students nested within classrooms was estimated. In that model, a statistically significant 34 percent of the variance in the post-assessment was attributable to classroom level variation.

Sets of covariates were added to the unconditional HLM model in this order:

Set 1. Pre-assessment score (standardized)

Set 2. Study Group (Bridge or Game Only)

Set 3. Student gender (1=Female)

Set 4. Classroom Level Characteristics: Whether or not they were enrolled in class in which more than half of the students completed the study (1=Yes); whether or not they were enrolled in an AP/Honors science class (1=Yes)

Set 5. In-game measures of implicit understanding—% Placement Errors, % Rotation Errors, and % Puzzle Errors (all standardized)

Set 6. Gameplay duration (>1 hour vs. not) and highest level reached (Level 22 vs. not)

Only statistically significant covariates were retained in the HLM model presented in this paper. Sets 3, 4, and 6 had no significant results, meaning student gender, Honors/AP status, gameplay duration and highest level reached were not significantly related to changes in pre-post assessment scores.

The model with the in-game measures of implicit understanding of slope and the Law of Reflection were a significantly better fit than the model without those measures (X^2 (3 df, N=317), 6.76, $p<0.10$). The best-fitting HLM model, which accounts for 33 percent of the variation at the classroom level, is presented in Table 3. Overall, after accounting for students' performance on the pre-assessment, students who exhibited more Placement and Rotation errors while playing the game performed more poorly on the post than students with lower science error rates.

Table 3: Best-fitting HLM model

Parameter	Est.	Std Err	df	Sig.	95% Confidence Interval	
					Lower	Upper
Intercept	0.10	0.12	24	0.43	-0.15	0.35
Pre-Assessment[1]	0.35	0.05	320	0.00	0.26	0.45
Bridge (vs. Game Only)	-0.17	0.17	25	0.33	-0.52	0.18
%Placement Errors[1]	-0.08	0.05	304	0.09	-0.17	0.01
%Rotation Errors[1]	-0.17	0.05	320	0.00	-0.26	-0.07
%Puzzle Errors[1]	0.00	0.04	310	0.93	-0.09	0.08

[1]Standardized

The intercept coefficient represents the estimated outcome for male students who scored at the mean level of the pre-assessment, were in the Game Only group, were not in a Honors/AP class, and had mean levels of Placement and Rotation Errors. These students would score 0.07 standard deviations below the mean post-assessment score. The Pre-Assessment coefficient reflects the change in number of standard deviations of the post-assessment for every increase of 1 standard deviation on the pre-assessment. For every standard deviation increase on the pre-assessment, students would be expected to score 0.35 standard deviations higher on the post-assessment. Students in Bridge classes scored 0.17 standard deviations lower on the post–assessment than students in Game Only classes—a non-significant difference. There was no significant difference between Bridge and Game Only groups in their pre-post gains. This may be because Game

Only classroom instruction provided lab experiences with lasers that mirrored what Bridge classrooms did with *Quantum Spectre*, providing comparable experiences and similar gains.

Students whose placement or rotation error rate was one standard deviation above the mean, however, had post-assessment scores 0.08 and 0.17 standard deviations below the mean, respectively. There was no impact of puzzle errors. Interactions between study group (Bridge vs. Game Only) and gameplay errors were examined but none significantly improved the fit of the HLM model, suggesting the impact of these errors was the same across study groups.

8. DISCUSSION & IMPLICATIONS

Hierarchical linear modeling suggest a direct negative relationship between science-related gameplay errors and implicit science learning—players making errors consistent with a lack of implicit science understanding performed worse than players not making as many of those errors. Educators can use this information as a real-time, or reflective, formative assessment tool. This could be very useful in a class where students are playing a learning game, individually or in groups, while the teacher has an app that alerts them to which students are struggling and may need attention. A more comprehensive dashboard they can use after class might show them overall progress of their class and trends that inform how the next lessons are planned. Teachers might also use a dashboard to monitor their students' game-based learning as they play at home or with friends outside of class. The ability to validly infer implicit science learning from the digital records of game activity makes this all possible.

9. ACKNOWLEDGMENTS

We thank the teachers and students who participated in this study. This research was funded as part of a NSF DRK12 grant 1119144 to develop and study the Leveling Up games. We gratefully acknowledge the rest of the EdGE team: Erin Bardar, Barbara MacEachern, Jamie Larsen, and Katie Stokinger for their design and outreach efforts.

10. REFERENCES

[1] Eagle, M., Rowe, E., Hicks, A., Brown, R., Barnes, T., Asbell-Clarke, J., & Edwards, T. (2015, October). Measuring implicit science learning using networks of player-game interactions. Presented at the annual ACM Symposium on Computer-Human Interaction in Play, London.

[2] Lenhart, A., Purcell, K., Smith, A., & Zickuhr, K. (2010). *Social Media & Mobile Internet Use Among Teens and Young Adults.* Washington, DC: Pew Research Center.

[3] National Research Council (2011). *Learning Science Through Computer Games and Simulations.* M.A. Honey and M.L. Hilton (Eds.), Wash., DC: National Academies Press.

[4] Shute, V. & Ventura, M. (2013). *Stealth assessment: Measuring and supporting learning in video games*. MIT Press.

[5] Steinkuehler, C., & Duncan, S. (2008). Scientific Habits of Mind in Virtual Worlds. *Journal of Science Education and Technology, 17*(6), 530–543.

[6] Eagle, M., Peddycord, B., Hicks, A., Barnes, T. (2015, April). Exploring networks of problem-solving interactions. In *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge* (LAK '15), Poughkeepsie, NY, USA, pp. 21-30.

[7] Eagle, M., Barnes, T. (2014, July). Exploring differences in problem solving with data-driven approach maps. In proceedings of *Educational Data Mining (EDM2014)*, London, UK, pp. 76-83.

[8] Polanyi, M. (1966). *The Tacit Dimension*. London: Routledge. (University of Chicago Press. ISBN 978-0-226-67298-4. 2009 reprint).

[9] Vygotsky, L.S. (1978). *Mind in society: The development of higher psychological processes.* Cambridge, Mass.: Harvard University Press.

[10] McCloskey, M. (1983). Intuitive Physics. *Scientific American, 248*(4), 122–130.

[11] Minstrell, J. (1982). Explaining the "at rest" condition of an object. *The physics teacher*, *20*(1), 10–14.

[12] diSessa, A.A. (1993). Toward an Epistemology of Physics. Cognition and Instruction, 10(2/3), 105–225. doi: 10.2307/3233725.

[13] GlassLab (2014). Psychometric Considerations In Game-Based Assessment. Institute of Play. From: http://www.instituteofplay.org/work/projects/glasslab-research/

[14] Shute, V., Ventura, M. & Kim, J. (2013). Assessment and Learning of Qualitative Physics in Newton's Playground. *The Journal of Educational Research*, *106 (6.*), 423–430, **doi:10.1080/00220671.2013.832970.**

[15] Clark, D.B., Nelson, B., Chang, H., D'Angelo, C.M., Slack, K. & Martinez-Garza, M. (2011). Exploring Newtonian mechanics in a conceptually-integrated digital game: Comparison of learning and affective outcomes for students in Taiwan and the United States. *Computers and Education,* 57(3), 2178–2195.

[16] Thomas, D., & Brown, J.S. (2011). *A New Culture of Learning: Cultivating the Imagination for a World of Constant Change*. Lexington, KY: CreateSpace.

[17] Plass, J., Homer, B.D., Kinzer, C.K., Chang, Y.K., Frye, J., Kaczetow, W., Isbister, K., & Perlin, K. (2013). Metrics in Simulations and Games for Learning. In M. Seif El-Nasr, A. Drachen, & A. Canossa (Eds.), *Game Analytics: Maximizing the Value of Player Data* (694-730). Springer-Verlag.

[18] Feng, M., & Heffernan, N.T. (2006). Informing teachers live about student learning: Reporting in the assistment system. *Technology Instruction Cognition and Learning*, *3*(1/2), 63.

[19] Lederman, L. C., & Fumitoshi, K. (1995). Debriefing the Debriefing Process: A new look. In D. C. K. Arai (Ed.), *Simulation and gaming across disciplines and cultures*. London: Sage Publications.

[20] Andres, J.M.A.L., Andres, J.M.L., Rodrigo, M.M.T., Baker, R.S., & Beck, J.B. (2015) An investigation of eureka and the affective states surrounding eureka moments. To appear in *Proceedings of the 23rd International Conference on Computers in Education*.

[21] Hicks, A., Eagle, M., Rowe, E., Asbell-Clarke, J., Edwards, T., & Barnes, T. (2016, April). Using game analytics to evaluate puzzle design and level progression in a serious game. Paper presented at the 6th international Learning Analytics & Knowledge conference, Edinburgh, U.K.

Assessing Student-Generated Design Justifications in Engineering Virtual Internships

Vasile Rus, Dipesh Gautam,
Department of Computer Science
Institute for Intelligent Systems
The University of Memphis
Memphis, TN 38152
{vrus,dgautam}@memphis.edu

Zachari Swiecki, David W. Shaffer
Department of Educational Psychology
University of Wisconsin-Madison
Madison, WI 53706
{swiecki,dws}@wisc.edu

Arthur C. Graesser
Department of Psychology
Institute for Intelligent Systems
The University of Memphis
Memphis, TN 38152
graesser@memphis.edu

ABSTRACT

Engineering virtual internships are simulations where students role play as interns at fictional companies, working to create engineering designs. To improve the scalability of these virtual internships, a reliable automated assessment system for tasks submitted by students is necessary. Therefore, we propose a machine learning approach to automatically assess student generated textual design justifications in two engineering virtual internships, *Nephrotex* and *RescuShell*. To this end, we compared two major categories of models: domain expert-driven vs. general text analysis models. The models were coupled with machine learning algorithms and evaluated using 10-fold cross validation. We found no quantitative differences among the two major categories of models, domain expert-driven vs. general text analysis, although there are major qualitative differences as discussed in the paper.

Keywords

Virtual internships, machine learning, auto-assessment, epistemic frame theory

1. INTRODUCTION

In virtual internships, students play the role of interns in a virtual training environment. In engineering virtual internships, such as *Nephrotex* (NTX) and *RescueShell* (RS), students research and create multiple engineering designs [1]. As part of their design process, they regularly submit written work in the form of electronic engineering notebooks that are assessed by human judges. This human assessment is labor intensive, time consuming, and error-prone under certain circumstances such as time pressure. Furthermore, prior work has suggested that the reliability of human assessments can vary depending on the traits of the assessor, their experience, and the types of problems being assessed [14]. Thus, an automated assessment method that could provide efficiency in terms of time and cost as well as improved reliability is much needed. Our work presented here constitutes a step in this direction.

In the present study, we explored various models for automatically assessing notebooks in the engineering virtual internships NTX and RS. The content of these notebooks varies; however, in this study we focus on only one type of notebook in which students must justify their engineering designs by typing a short, free-text justification.

We have experimented with models that emulate an expert analysis of the student notebook entries as well as models derived from general textual analysis features. It should be noted that our work differs from previous attempts which rely on a semantic similarity approach, i.e. measuring how semantically close a student-generated response is to an ideal, expert-generated response as in [6].

The domain expert-driven models incorporate theoretically driven, content-based features identified by human experts such as "referencing any performance parameter such as cost", which is a general design feature because it applies to all engineering designs in NTX and RS, or "indicating the power source", a feature specific to the concrete task of designing an exoskeleton, which was the focus of the RS internship and not NTX. A challenge with the domain expert-driven models is that the features are specific to either the type of task, e.g. engineering design, or the concrete task itself, e.g. design an exoskeleton. This results in a scalability issue as these models must be redesigned manually by domain experts when moving to a new domain, new type of task, and/or a new concrete task. However, the net theoretical advantage of these domain expert-driven models is that they are tailored to the task at hand and therefore are expected to yield very good performance. These models also afford the ability to create automatic and tailored feedback to students given their task-specific diagnostic capabilities.

The other category of models that we used rely on general text analysis features inspired from previous work on automated essay scoring [2,5,13] and text analysis software tools such as Coh-Metrix [4] and LIWC [7]. For instance, in automated essay scoring the length in words of the essay, i.e. the number of all word occurrences or word tokens, is by far the best predictor of essay quality. Coh-Metrix is a software package that calculates the coherence of texts in terms of co-reference, temporal cohesion, spatial cohesion, structural cohesion, and causal/intentional cohesion. LIWC (Linguistic Inquiry and Word Count) uses a word count strategy to characterize texts along a number of dimensions that include standard language categories (e.g., articles, prepositions, pronouns), psychological processes (e.g., positive and negative emotion word categories), and traditional content dimensions (e.g., sex, death, home, occupation).

The key advantage of the general text analysis models is that they are generally applicable across types of tasks, specific tasks, and domains. In addition, the general text analysis features are relatively cost-effective and easy to derive from the data compared

to features derived by domain experts, which require (significantly more) human time and effort.

In this paper, we explore the predictive power of the two major categories of models mentioned above, domain-expert vs. general text analysis, in conjunction with a number of machine learning algorithms such as decision trees, naïve Bayes, Bayes Nets, and logistic regression. Furthermore, we employed an ensemble of classifiers approach in order to boost the performance of individual models. We conclude the paper with a qualitative assessment of the relative benefits of the proposed models for virtual internships by considering their predictive value, the labor involved in their development, and their ability to provide interpretable assessments for students.

2. BACKGROUND

We review in this section prior work on assessing students' open-ended responses with an emphasis on prior work in the area of educational technologies.

Automated essay scoring systems [2,5,13] have been developed for more than two decades as a way to tackle the costs, reliability, generality, and scalability challenges associated with assessing student generated open-ended responses to essay prompts. There are a number of systems available for automated essay scoring, some of which are commercial. It is beyond the scope of this paper to offer a thorough review of the work in this area. We limit ourselves to noting that the focus on automated essay scoring is on the argumentative power of an entire essay while in our case the focus is on required (design) items that must be present in paragraph-like justifications. This entails that style and higher-level constructs such as rhetorical structure are less important in our task as opposed to the essay scoring task and that factors that focus more on content measures are highly important. Given these differences and the fact that the two most predictive factors of essay quality are also content related, we included in our models the following two features: word count, i.e. total number of word occurrences or tokens in student justifications, and content word count, i.e. the total number of content word occurrences (nouns, verbs, adjectives, and adverbs).

Directly relevant to our study is previous work by Rus, Feng, Brandon, Crossley, and McNamara [8] who studied the problem of assessing student-generated paraphrases in the context of a writing strategy training tutoring system. One of the strategies in this tutoring system is paraphrasing. As the system is supposed to prompt students to paraphrase and then provide feedback on their paraphrases, Rus and colleagues collected a large corpus of student-generated paraphrases and analyzed them along several dozen linguistic dimensions ranging from cohesion to lexical diversity obtained from Coh-Metrix [4]. There are significant differences between their work and ours. First, we deal with justifications which can vary in length from a few words to a full paragraph as opposed to explicitly elicited paraphrases of target sentences. Second, we do use extra features to build our models besides the Coh-Metrix indices. Third, we assess the student generated justifications as acceptable or unacceptable (i.e., correct or incorrect). We could eventually investigate finer levels of correctness, e.g. on a scale from 1-5, which we plan to do as part of our future work.

Williams and D'Mello [15] worked on predicting the quality of student answers (as error-ridden, vague, partially-correct or correct) to human tutor questions, based on dictionary-based dialogue features previously shown to be good detectors of cognitive processes (cf. [15]). To extract these features, they used LIWC (Linguistic Inquiry and Word Count; [6]), a text analysis software program that calculates the degree to which people use various categories of words across a wide array of texts genres. They reported that pronouns (e.g. I, they, those) and discrepant terms (e.g. should, could, would) are good predictors of the conceptual quality of student responses. Like Williams and D'Mello, we do use LIWC to analyze student notebooks' justifications. Furthermore, we employ expert-identified features and features from Coh-Metrix and automated essay scoring.

Prior work by Rus, Lintean, and Azevedo [9] investigated the performance of several automated models designed to infer the mental models of students participating in an intelligent tutoring system (ITS). The ITS was designed to teach students self-regulatory processes while they were learning about science topics such as the human circulatory system. Rus and colleagues used two methods, a content-based method and a word-weighting method, to derive features for their models. While our present work does not investigate models using word-weighting methods, we do investigate models using content-based features.

The content-based features used by Rus and colleagues included a taxonomy of relevant biology concepts derived by human experts, expert annotated pages of content from the ITS, and expert-generated paragraphs. In the present study, the content-based features, or domain-expert (DE) features, we used consist of discourse codes developed by human experts. Discourse codes indicate the presence or absence of specific concepts in student talk, or in this case, student written work. The DE features were developed through a grounded analysis of student design justifications collected from engineering virtual internships [3].

The learning that occurs in engineering virtual internships can be characterized by epistemic frame theory. This theory claims that professionals develop epistemic frames, or the network of skills, knowledge, identity, values, and epistemology that are unique to that profession [11]. For example, engineers share ways of understanding and doing (knowledge and skills); beliefs about which problems are worth investigating (values), characteristics that define them as members of the profession (identity), and a ways of justifying decisions (epistemology). In this study, we used epistemic frame theory to guide the development of the DE features. In prior work, elements of the engineering epistemic frame have been operationalized as discourse codes and used to assess engineering thinking in virtual internships [1]. In this study, the DE features we identified correspond to elements of the engineering epistemic frame that relate to justifying design decisions. The presence or absence of these features in a student's written work thus represents elements of the engineering epistemic frame that are present or lacking.

In sum, we used some of the features described by the above researchers in our work, such as word count, as well as novel features, e.g. features based on the engineering epistemic frame.

3. ENGINEERING VIRTUAL INTERNSHIPS

In this study, we examined student written work collected from the engineering virtual internships, *Nephrotex* (NTX) and *RescueShell* (RS). In NTX, students work in teams to design filtration membranes for hemodialysis machines, while in RS,

student teams design the legs of a mechanical exoskeleton used by rescue workers.

All interactions in virtual internships take place via a website in which students communicate with their teams using email and chat. During the internships, students research and create engineering designs in two cycles. In each cycle, students design five prototypes and later receive performance results for each prototype which they have to analyze and interpret.

During their design process, students submit records of their work via electronic notebook entries for each substantive task they complete, including summarizing research reports and justifying design decisions. The expectations of notebook entries are outlined in prompts, which students receive via email in the virtual internship website. Each notebook that students submit is divided into notebook sections, i.e., separate text fields for items that are defined by the email prompts. In this study, we analyzed notebook sections in which students provided justifications for their prototype design decisions.

Once students complete each notebook section, they submit the notebooks to trained human raters for assessment. In the fiction of the virtual internships, these raters play the role of more senior employees in the company who act as *mentors* to the students. The role of the mentors is to answer student questions and lead team discussions, in addition to assessing student work.

Once a mentor receives a notebook, they assess each section as acceptable or unacceptable using provided rubrics. The assessment system used by the mentors automatically generates pre-scripted feedback corresponding to the assessment given to each section. Currently, this feedback is generic in the sense that it does not respond to the particulars of a student's response. For example, an assessment of unacceptable on a notebook section requiring a summary generates feedback that (1) informs the student that the section was unacceptable, (2) reminds them of the content they were asked to summarize, and (3) points them to the documents they were asked to summarize. This automated feedback does not inform the student exactly why the section was rated as unacceptable. However, the mentor does have the option to compose specific feedback for the student if they wish.

Our work here moves us towards a more automated and student-tailored assessment and feedback mechanisms which could have significant impact on the economy of scaling virtual internships to all students, anytime, anywhere via Internet-connected devices.

4. EXPERIMENTS AND RESULTS

We describe first the data set we used in our experiments before presenting the experiments and results obtained with the models.

4.1 Data Set

In this study, we analyzed notebook sections from the NTX and RS virtual internships in which students justified their engineering design decisions. In these notebook sections, students were required to include the design input choices they selected—that is, their design specifications, and a justification explaining why this design was chosen for testing.

Mentors assessed these notebook entries as acceptable or unacceptable in real-time during the virtual internship using the following rubric:

1. Listed their design specifications
2. Included a justification referencing at least one design specification.

Acceptable justification may include:

1. Prioritizing attributes
2. Referencing internal consultant requests
3. The performance of a design specification on a specific attribute
4. Experimental justifications (e.g., holding design specifications constant)

To select data for this study, we randomly sampled 298 justification sections from 20 virtual internship sites, i.e. datasets corresponding to 20 schools where the virtual internships were implemented. Twelve were NTX sites and eight were RS sites. Of the 298 justifications sampled, 146 were from NTX and 152 were from RS. Students were given the same prompts for justification sections in NTX and RS. In addition, the same rubrics were used by raters in NTX and RS. Thus, we combined data from RS and NTX to train our models.

As described above, justification sections were originally assessed by mentors during the virtual internship in real time. The mentors were trained to assess notebook section, but they were not experts in the domain of engineering or the content of the virtual internships. In addition, they had to assess notebook sections under time constraints and while completing their other responsibilities as a mentor. For example, they could have to respond to student questions via chat while assessing. Thus, to obtain potentially more valid and reliable assessments for model training, the justification sections in this study were re-assessed by more experienced raters that did not face the constraints placed on the mentors. We found that the agreement between the human mentors and our experienced raters on the 298 student justifications we used in this work was kappa = 0.271. This value is very low, indicating that mentors' assessments are not reliable, as we suspected.

Each justification section was re-assessed by two new raters, benchmark rater 1 (BE1) and benchmark rater 2 (BE2). BE1 had over two years of experience rating notebook sections from virtual internships and had contributed to the content development of both NTX and RS. BE1 was thus considered an expert rater for the purposes of this study. BE2 was a less experienced rater trained to assess justification sections. BE1 and BE2 assessed all 298 justification sections using the rubric above and agreed on one final judgement (acceptable or unacceptable) for each justification. Their inter-annotator reliability as measured by kappa was 0.767**.** Table 1 includes examples of notebook sections from NTX assessed as acceptable and unacceptable by the benchmark raters. About 73% of the instances in the data set were rated positively by the BEs. The distribution of positive and negative instances is shown in Table 2.

4.2 Feature Selection

As already mentioned, we focused on two major categories of models: models that rely on domain-experts (DE) versus models that rely on more general textual analysis features. We developed the DE features through a grounded analysis [3] of a sample of 98 justification sections. These features were developed by two researchers who re-assessed the sample and developed discourse codes corresponding to what they attended to while assessing. Next, we automated these codes using the *nCoder*, a tool for developing and validating automated discourse codes that relies

on authoring targeted regular expressions for each of the expert-identified codes [12]. These codes were included as features in our models (see Table 3 for descriptions).

Table 1. Example of Acceptable and unacceptable notebooks from the virtual internship *Nephrotex*

Notebook entry	Assessment
Design Specifications: PAM, Vapor, Negative Charge, 4 % Justification: This prototype was altered slightly from the original with this material by changing from 2% CNT to 4%. This is an attempt to increase reliability without hindering flux or blood cell reactivity.	**Acceptable**
Design Specifications: PAM, Vapor, Negative Charge, 2.0 Justification: These specificaions ran best for PAM material	**Unacceptable**

Table 2. Distribution of human-ratings in the 298 instances.

Human Rating	#Instances
Acceptable	217
Unacceptable	81
Total	298

The general textual analysis features were further divided by their source into the following three categories: features inspired from automated essay scoring (ES) research, features obtained with the automated tool for textual analysis Coh-Metrix, and features obtained with the automated tool for textual analysis LIWC. This categorization of the general textual analysis features is needed for several reasons. First, the various sources capture different aspects of a text. Second, this categorization allows us to conduct ablation studies in which we assess the contribution of each major category of features to solving the task at hand. It should be noted that there is overlap among the features from various groups/sources. For instance, the WC (LIWC), DESWC (Coh-Metrix), and Word_Count (DE) features are all counts of white-spaces in a target text, i.e. justifications in our case. These features are slightly different from the token Count feature in the ES group which counts number of tokens after applying the Stanford tokenizer tool. Similar features will not end up in the same models if they correlate highly, as explained next.

Not all features have equal predictive power and having redundant or irrelevant features can decrease the performance of the models. Therefore, we had a feature selection step keeping features that have low correlation with each other (<.70). When two features in a model had a correlation greater than .70 of them was dropped. For instance, from the LIWC and Coh-Metrix groups of features the features selected via this process were: WC, SIXLTR, adverbs, verbs, DESSC, DESSL, DESSLd, PCNARz, PCCONNp (See Table 3 for descriptions). The feature selection step was needed given that we worked with various machine learning algorithms, some of which do not have a feature selection process linked to them, e.g. the stepwise variable selection in some regression implementations.

4.3 Results

We experimented with the proposed models in conjunction with a number of classification algorithms including decision trees, naïve Bayes, Bayes Nets, and logistic regression. We present here the results obtained with the logistic regression classifier as it yielded the best results overall. The models were validated using 10-fold cross validation. Performance was measured using standard measures such as accuracy, false positive rate, precision, recall, F-measure, and kappa statistic. The false positive rate, the percentage of true negatives predicted as positives, is of special interest because it gives us an idea of how many justifications are deemed correct when in fact are not, by a particular method. That is, it indicates how many opportunities for feedback a specific method might miss as a justification deemed correct means there is no need for specific feedback to improve it. The evaluation results are shown in Table 4. We focus next on the most important model comparisons due to space constraints, e.g. we do not show results when combining two groups of features.

We started with models that included features from only one group, i.e. the individual feature group models shown in rows 1-4 in Table 4, selected the best such model and then added, sequentially, features from the other groups in batches, where each batch contained the selected features in one group. This procedure, also known as an ablation study in machine learning, allows to see what we gain if we add a group of features to a model that already contains feature from one or more groups. From Table 4, we infer that the ES and Coh-Metrix individual models are the best as they have slightly higher accuracy in prediction (85.23% for ES and 85.23% for Coh-Metrix) compared to other two individual feature groups. Also their kappas are the highest among the models with only one group of features.

In row 5, we show the results when combining all general text analysis features: ES, LIWC, and Coh-Metrix. As already mentioned before, we are directly interested in comparing the domain expert-driven model, derived from the DE features, with the model in row 5 that includes all the general text analysis features from the ES, LIWC, and Coh-Metrix groups. As we notice, these two qualitatively different models have very similar performance across all performance measures.

In addition to developing the above models from subsets of features, we used ensembles of 3 individual and combined models, respectively, in conjunction with a majority voting mechanism. For instance, if 2 or 3 out of 3 models predicted a justification as *accepted* then the final prediction for the instance was *accepted*. We experimented with voting in two different ways: (1) we used the best 3 models from the individual or combined groups of features; (2) we used the weakest 3 models obtained with any combinations of features from individual and combined groups of features; this latter case is based on results from statistics that show that combining weak classifiers should result, in general, in better performance relative to the performance of each of the weak classifiers. Both types of ensembles (weakest versus best) yielded in the best cases similar accuracies of ~86% and similar performance across all the other performance measures. The false positive rate of the weakest combined model ensemble was lowest.

5. CONCLUSIONS

In this paper, we experimented with multiple models designed to automatically assess notebook sections from engineering virtual internships. In particular, we developed models to assess notebook sections in which students justified design decisions. All models performed very well with good and very good kappa scores (kappas scores of 0.6-0.8 are considered very good)

Table 3. Descriptions of the some features used in the proposed models (not all shown due to space constraints).

Features	Description
	LIWC
Word Count	*Word Count* (WC; Total number of words in text), *Token Count* (TC; Number of unique words in text), *Words > 6 letters* (SIXLTR: total number of words greater than 6 letters) *Punctuations*
Type Token Ratio	*Ratio of TC and WC*
	Coh-Metrix
Lexical Component Counts	*DESPC* - Paragraph count, number of paragraphs; DESSC - Sentence count, number of sentences, DESWC - Word count, number of words
DESPL	DESPL - Paragraph length, number of sentences, mean; DESPLd - Paragraph length, number of sentences, standard deviation; DESSLd; Sentence length, number of words, standard deviation;
Connectives Features	PCCONNp - the degree to which the text contains connectives such as adversative, additives and comparative connectives to express relations in the text.
Temporality Features	PCTEMPz - the temporality such as tense or aspect of the text; SMTEMP - temporal cohesion, measured by repetition score of tense and aspect
LDTTRa	Type token ratio of all words.
	Domain Expert (DE)
Exoskeleton Design Inputs	Control Sensor, Range of Motion, Power Source, Material, Actuator
Dialyzer Design Inputs	Process, Surfactant, Material, Carbon Nanotube Percentage
Attributes	Referencing any design attribute or performance parameter such as cost, reliability, etc.
Justification Features	*Balancing* - Justifying input choices by stating it made up for the weakness of another choice or by saying that another choice will balance out its weaknesses; *Client* - Justifying input choices by stating it would be good for the client or end user of the product; *Consultant.Requests* - Justifying input choices because the results meet or are expected to meet internal consultants' requests; *Evaluation* - Justifying input choices by evaluating the performance of the inputs
	Essay Scoring (ES)
Token Count	Count of word occurrences in the justification.
Content Word Count	Count of all content words (noun, adjective, verb, adverb) in the justification.

Table 4. Performance evaluation results for various models.

S.N.	Features	Accuracy	FP Rate	Precision	Recall	F-Measure	Kappa
1	ES	85.2349	0.2490	0.850	0.8520	0.8510	0.6181
2	LIWC	83.2215	0.2950	0.8270	0.832	0.8290	0.5591
3	Coh-Metrix	85.2349	0.2950	0.8480	0.8520	0.8460	0.5991
4	DE	83.2215	0.3020	0.8270	0.8320	0.8280	0.5555
5	ES+LIWC+Coh-Metrix	83.8926	0.2920	0.8340	0.8390	0.8350	0.5733
6	LIWC + DE + Coh-Metrix + ES	81.8792	0.3000	0.8150	0.8190	0.8170	0.5314

indicating that they are much better than chance predictions. Our results show that, in this context, the predictive value of models using only the general text analysis features is comparable to the predictive value of a model using only the DE features (a McNemar's test on paired nominal data revealed no significant difference between the two models' prediction).

In particular, the ES group of features is the best predictor of students' justifications quality. When other groups of features are added to the individual ES model, the results do not improve significantly. The fact that the ES features are so good is not surprising. Word count, or essay length, which is one of the features in the ES group, is known as being the best predictor of essay quality in automated essay grading [6,10]. Also, the Coh-Metrix group of features are a good predictor of the quality of students' justifications.

It is important to note, however, that the predictive power of a model is only one dimension for evaluating the utility of automated assessment models in learning environments like virtual internships. We suggest that developmental cost and interpretability of the models are also valuable dimensions to

consider. Of the models presented above, those using only the general text analysis features have the lowest developmental cost. Moreover, these features are generally applicable across types of tasks, specific tasks, and domains. In contrast, models containing the DE features have a relatively high developmental cost because their features required the time and expertise of humans to develop. We do note that the DE features described in this paper were automated. Thus, they can readily be applied to more justification sections from engineering virtual internships. However, these DE features are specific to this context and are likely not generalizable outside of engineering virtual internships.

The utility of these automated assessment models lies in implementing them in real-time during a virtual internship where they will be used to assess student work and either generate automatic feedback or suggest feedback for human mentors to give. For the models using only the general text analysis features, any potential feedback would be in terms of features such as word count or "narrativity" of the text that are not directly related to the domain-relevant content of the text. Those models using DE features, however, could potentially generate domain-relevant feedback in terms of what DE features were present and absent in the text. For example, if a student's justification section fails to relate their design decisions to the requests of the company's internal consultants, that is, it lacks the "Consultant Requests" DE feature, feedback could be suggested to the mentor or provided automatically to the student informing them of this missing information and suggesting ways to include it. Thus, in terms of ease of interpretation, those models using only the general text analysis features have a relatively low ease of interpretation compared to those models that include the DE features.

In this context, we then suggest the use of the best predictive model to assess the overall quality of justifications in engineering virtual engineering internships, and subsequently use the DE-based model to identify potential domain-specific missing parts in an unacceptable justification in order to provide direct feedback to the student or at least make suggestions to human mentors regarding possible weak aspects of the justification. This approach balances the tradeoffs between generality and reliability versus domain and task specific diagnostic capabilities.

We plan to further improve the predictive power, generality, and diagnostic capabilities of our models. For instance, we are considering unsupervised methods to automatically detect domain specific codes that could be used as features in our DE models. Furthermore, we are considering unsupervised topic detection in student-generated justification as a way to generalize the applicability of our models to other domains and types of tasks.

6. ACKNOWLEDGMENTS

This work was funded in part by the National Science Foundation (DRL-0918409, DRL-0946372, DRL-1247262, DRL-1418288, DUE-0919347, DUE-1225885, EEC-1232656, EEC-1340402, REC-0347000), the MacArthur Foundation, the Spencer Foundation, the Wisconsin Alumni Research Foundation, and the Office of the Vice Chancellor for Research and Graduate Education at the University of Wisconsin-Madison. The opinions, findings, and conclusions do not reflect the views of the funding agencies, cooperating institutions, or other individuals.

7. REFERENCES

[1] Chesler, N. C., Ruis, A. R., Collier, W., Swiecki, Z., Arastoopour, G., & Shaffer, D. W. (2015). A novel paradigm for engineering education: Virtual internships with individualized mentoring and assessment of engineering thinking. *Journal of biomechanical engineering*, *137*(2).

[2] Dikli, S. (2006). An Overview of Automated Scoring of Essays. *Journal of Tech., Learning, and Assessment*, 5(1).

[3] Glaser, B. G., & Strauss, A. L. (2009). The discovery of grounded theory: Strategies for qualitative research. Transaction Publishers.

[4] Graesser, A. C., McNamara, D. S., Louwerse, M. M., and Cai, Z. (2004). Coh-Metrix: Analysis of text on cohesion and language. *Behavior research methods, instruments, & computers*, 36, 2(2004), 193-202.

[5] Leacock, C., and Chodorow, M. 2003. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37, 4(2003), 389-405.

[6] Mohler, M., and Mihalcea, R. 2009. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the EACL* (Athens, Greece, March, 2009).

[7] Pennebaker, J. W., Francis, M. E., and Booth, R. J. 2001. Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates*, 71, (2001), 2001.

[8] Rus, V., Feng, S., Brandon, R., Crossley, S., and McNamara, D.S. (2011). A Linguistic Analysis Of Student Generated Paraphrases. In R. C. Murray and P.M. McCarthy (Eds.), Proceedings Of The 24th International Florida Artificial Intelligence Research Society Conference. Menlo Park, CA: AAAI Press.

[9] Rus, V., Lintean, M., Azevedo, R. (2009). Automatic Detection of Student Mental Models During Prior Knowledge Activation in MetaTutor. Proceedings of the 2nd International Conference on Educational Data Mining. Cordoba, Spain.

[10] Rus, V., Niraula, N. (2012). Automated Detection of Local Coherence in Short Essays Based on Centering Theory", CICling 2012, March 11-17, IIT Delhi, India.

[11] Shaffer, D. W. (2006). *How computer games help children learn*. Macmillan.

[12] Shaffer, D.W., Borden, F., Srinivasan, A., Saucerman, J., Arastoopour, G., Collier, W., Ruis, A.R., & Frank, K.A. (2015). The *nCoder*: A technique for improving the utility of inter-rater reliability statistics. Epistemic Games Group Working Paper 2015-01. University of Wisconsin–Madison.

[13] Shermis, M.D. & Burstein, J. (2003). Automated Essay Scoring: A Cross Disciplinary Perspective. Lawrence Erlbaum Associates, Mahwah (2003).

[14] Tisi, J., Whitehouse, G., Maughan, S., & Burdett, N. (2013). *A review of literature on marking reliability research* (Report for Ofqual). Slough: NFER.

[15] Williams, C., &D'Mello, S. (2010). Predicting student knowledge level from domain-independent function and content words. In Intelligent Tutoring Systems (pp. 62-71). Springer Berlin Heidelberg.

Tensor Factorization for Student Modeling and Performance Prediction in Unstructured Domain

Shaghayegh Sahebi
Intelligent systems Program
University of Pittsburgh
Pittsburgh, PA
shs106@pitt.edu

Yu-Ru Lin
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA
yurulin@pitt.edu

Peter Brusilovsky
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA
peterb@pitt.edu

ABSTRACT

We propose a novel tensor factorization approach, Feedback-Driven Tensor Factorization (FDTF), for modeling student learning process and predicting student performance. This approach decomposes a tensor that is built upon students' attempt sequence, while considering the quizzes students select to work with as its feedback. FDTF does not require any prior domain knowledge, such as learning resource skills, concept maps, or Q-matrices. The proposed approach differs significantly from other tensor factorization approaches, as it explicitly models the learning progress of students while interacting with the learning resources. We compare our approach to other state-of-the-art approaches in the task of Predicting Student Performance (PSP). Our experiments show that FDTF performs significantly better compared to baseline methods, including Bayesian Knowledge Tracing and a state-of-the-art tensor factorization approach.

Keywords

Tensor factorization, student modeling, predicting students performance, learning analytics

1. INTRODUCTION

The growth of Massive Open Online Courses (MOOC) has rapidly increased the volume of data on students' education and learning behavior. This abundance of data calls for approaches that can automatically make sense of such data, and that remove the need for manual handling of such massive amounts of data. Predicting students performance and modeling student knowledge are two of the tasks that help researchers to understand such data. The goal in predicting student performance (PSP), is to estimate if a specific target student can handle a learning material successfully – for example, whether the student can succeed or fail at solving a specific quiz. Student knowledge modeling aims to quantify or infer a student's knowledge at each moment in time in each of the possible skills (or concepts) the student may have. The set of skills are defined either manually or automatically based on the learning materials.

Understanding students' attempt data through PSP and student knowledge modeling encourages teachers to design better courses, allows for targeted personalization of course pace, and provides more accurate automatic learning material recommendation to students. Hence, a primary focus in educational data mining literature is on predicting student performance and student knowledge modeling. For example, Bayesian Knowledge Tracing was one of the pioneering approaches that could predict the success or failure of students in solving problems [1].

Recently, other approaches, such as factorization models, have been used for PSP. For example, Performance Factor Analysis (PFA) [5] is another approach to PSP and cognitive modeling. PFA takes into account the effects of the initial difficulty of the skills (knowledge components) and prior successes and failures of a student at learning the skills associated with the current item. These approaches require prior knowledge of the overall domain model – the association between skills and learning material.

More recent approaches have sought to overcome this limitation by using latent factor approaches. For example, Thai-Nghe et al. experimented on a context-aware factorization algorithm, based on collaborative filtering approaches, in the relevant recommender system literature [9]. Sahebi et al. studied various methods of the educational data mining field with matrix and tensor factorization approaches, from the recommender systems literature for PSP [7]. Lan et al. used quantized matrix completion to predict students' performance in SPARFA-Lite [4]. This method solves a convex optimization problem and gives a global optimum solution.

Tensors, or multi-dimensional arrays, have been used in the literature to represent data on student attempts [6]. One of the main reasons that tensors are a suitable representation for modeling educational data is their seamless integration ability and flexibility in representing multiple dimensions of the data, such as students, questions, time, and topic structure. Another reason for using tensors is their capability for decomposing interactions in multi-dimensional data.

While various tensor decomposition models and algorithms already exist in the literature [3], the potential for versa-

tile modeling of tensors in the educational data mining field is under-explored. Although previous tensor factorization models that have been used in the literature have resulted in comparable performance in the task of PSP [6, 8], they are not tailored to educational data. More specifically, these models are built for purposes other than educational data mining (such as recommender systems), and thus do not consider the characteristics of educational data mining challenges.

One of these challenges is increases in student knowledge that occurs while they interact with learning material. As the students learn through quizzes, readings, and other learning resources, they incrementally learn the underlying skills that are present in these resources. Thus, this amount of knowledge increase for a student depends on the material that the student is interacting with. The current tensor factorization approaches that are used for PSP in the literature do not model this interaction.

In this paper, we provide a solution to this problem by proposing a unique tensor factorization-based approach that can account for the constant learning of students. Our proposed tensor factorization model, called *feedback-driven tensor factorization*, directly models the increases in student knowledge by adding a feedback-based constraint on the previous student's knowledge and the current learning material that a student is using. We compare our approach to Bayesian Knowledge Tracing and a baseline tensor factorization algorithm. Our experiments show the superior performance of our proposed approach, as compared to the baseline methods.

2. FEEDBACK-DRIVEN TENSOR FACTORIZATION (FDTF)

As mentioned in the introduction, the goal of our approach is to predict student performance while considering the fact that students are constantly learning. In order to achieve this goal, we represent student activities on learning material as a three-dimensional tensor $\mathcal{Y}$.

Notations. In this paper, tensors are represented by script letters, e.g. $\mathcal{Y}$; Matrices are denoted by boldface capital letters, e.g. $\mathbf{X}$; and vectors are represented by boldface lowercase letters, e.g. $\mathbf{x}$. In addition, we denote the i^{th} row of a matrix $\mathbf{X}$ as $\mathbf{X}_{i,:}$, the j^{th} column as $\mathbf{X}_{:,j}$, and the entry (i, j) as $\mathbf{X}_{i,j}$.

Suppose that students are working with one resource type and are learning from it. To be more specific, suppose that m students are interacting with n quizzes, and that each student can have multiple attempts (at most l) on each quiz. Then, we can represent the students' attempt sequences on all quizzes as a tensor of size $m \times n \times l$. The k^{th} frontal slice of this tensor ($\mathcal{Y}_{:,:,k}$) shows the success or failure of all students on all quizzes in their k^{th} attempt. To abbreviate, we use $\mathcal{Y}_k$ to represent the k^{th} frontal slice of all tensors. Accordingly, $\mathcal{Y}_{i,:,:}$ shows all the attempts of student i on all questions and $\mathcal{Y}_{:,j,:}$ shows all attempts of all students on question j. We assume that each quiz consists of multiple (c) concepts (skills or knowledge components) and that the students should have some knowledge of these concepts in order to solve the quizzes that include such concepts. Some

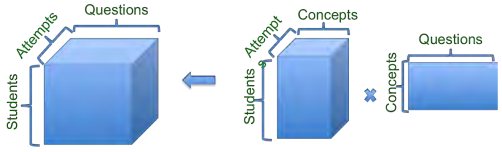


Figure 1: Phase 1: Decomposition of Student Performance into Student Knowledge and Concept-Map

of the elements of $\mathcal{Y}$ are unknown to us because not all of the students try all of the questions as many times. Based on these assumptions, we formulate the problem as a tensor factorization with two phases: the *prediction* phase and the *learning* phase.

In the prediction phase, we follow the assumption that students' success or failure in quizzes depends on their knowledge and the concepts underlying those quizzes. In this phase, we decompose $\mathcal{Y}$ into a tensor and a matrix: the tensor $\mathcal{T}$ that shows the knowledge of students on the concepts at each of their attempts on the quizzes, and the matrix $\mathbf{Q}$ that shows the concepts that are required to solve each quiz correctly. For each quiz j, $\mathbf{Q}_{:,j}$ shows the importance of each of the discovered concepts in it. Also, $\mathcal{T}_{i,k,l}$ shows the knowledge of student i in concept k at the l^{th} attempt.

Based on this decomposition, we can estimate (predict) the unknown values of $\mathcal{Y}$ using the multiplication of tensor $\mathcal{T}$ and matrix $\mathbf{Q}$, as presented in Equation 1. Figure 1 gives an illustration of this decomposition.

$$\mathcal{Y} = \mathcal{T} \times \mathbf{Q} \tag{1}$$

We suppose that students learn by practicing the quizzes, and that the knowledge of students increases through this practice of the concepts. The learning phase of our tensor factorization approach models student learning, based on the quizzes that they choose to solve in each step. In order to do that, we construct a tensor $\mathcal{X}$ that denotes when a student has or has not chosen to work on a specific problem at a specific time. Equation 2 shows how to build this tensor, based on $\mathcal{Y}$.

$$\mathcal{X}_{i,j,k} = \begin{cases} 1, & \text{if } \mathcal{Y}_{i,j,k} \text{ is observed} \\ 0, & \text{if } \mathcal{Y}_{i,j,k} \text{ is not observed} \end{cases} \tag{2}$$

In the learning phase, we assume that the amount of gained knowledge in each concept is a function of the student's knowledge at the previous attempt, as well as the weight of concepts that are learned in the quiz that the student chooses to solve. Let $f(\cdot)$ be such a function; then the gained knowledge at time t can be expressed as:

$$\mathcal{T}_t = f(\mathcal{T}_{t-1}, \mathcal{X}_t, \mathbf{Q})$$

Since we assume that knowledge of students grows over time, we should choose a monotonically increasing function for

$f(\cdot)$. Also, to keep this knowledge increase from growing too large, this function should be bounded. Based on these assumptions, we model the knowledge growth of students as a logistic regression function that ranges between 0 (for no increase in the knowledge) to $1 - \mathcal{T}_{t-1}$ (for a maximum increase in the knowledge). This allows us to have a bounded amount of knowledge that always stays between zero and one. To add to the flexibility of this function, and to account for different students' rate for learning from the quizzes, we add a factor μ that controls the slope of the logistic regression function. The higher the learning rate (μ), the larger the knowledge increase and the faster the students reach a maximum state of knowledge. This increase can be seen in Equation 3.

$$\mathcal{T}_t = \mathcal{T}_{t-1} + (\frac{2(1 - \mathcal{T}_{t-1})}{1 + exp(-\mu \mathcal{X}_t \mathbf{Q}')} - (1 - \mathcal{T}_{t-1})), \quad (3)$$

which can be written as follows:

$$\mathcal{T}_t = 2\mathcal{T}_{t-1} + \frac{2(1 - \mathcal{T}_{t-1})}{1 + exp(-\mu \mathcal{X}_t \mathbf{Q}')} - 1 \quad (4)$$

Based on this model, the more knowledgeable the student is in a concept, the less improvement she will obtain by practicing the same concepts again and again. The greatest increase in the student's knowledge happens when the student does not know the skills that are provided in the quiz. If we expand and simplify Equation 3, we achieve Equation 4. Since $f(\cdot)$ is a monotonically increasing function, the estimated knowledge tensor ($\mathcal{T}$) and domain model ($\mathbf{Q}$) are both non-negative. This non-negativity is in accordance with assumptions in the educational domain: that the weight of each concept in each learning material cannot be negative and that the knowledge of students at any time and in any concept cannot be negative either.

Eventually, the matrix factorization includes solving Equations 1 and 4. Assuming that we have the values for $\mathcal{X}_t$ and $\mathbf{Q}$, Equation 4 can be considered as a static update and we can only optimize Equation 1 iteratively and update the knowledge values in each iteration using Equation 4. To achieve this goal, we try to optimize for the least regularized estimation error of our observed tensor ($\mathcal{Y}$) in Equation 5. Thus, our objective is to minimize the overall error, which is defined as:

$$\Sigma_{i=1}^{t} \parallel \mathcal{Y}_t - \mathcal{T}_t \mathbf{Q} \parallel^2 + \lambda(\Sigma_{i=1}^{t} \parallel \mathcal{T}_i \parallel^2 + \parallel \mathbf{Q} \parallel^2), \quad (5)$$

where λ is a regularization parameter. The last two terms are added to the error equation to regularize the values in tensor $\mathcal{T}$ and matrix $\mathbf{Q}$. These two terms increase the sparsity of the knowledge and domain model by decreasing the values in these two factors, while preventing the factorization from being over-fit to the training data.

Since this method uses the iterative feedback loops and the two phases of prediction and learning, we name it Feedback-Driven Tensor Factorization (FDTF).

3. EXPERIMENTS

To asses the student performance prediction task, we compare the proposed FDTF model to a baseline tensor factorization algorithm that was introduced in previous recommender system literature. This tensor factorization algorithm is called the Bayesian Probabilistic Tensor Factorization (BPTF) and models the temporal change of user interests on items [10]. We choose this model as a baseline because of its consideration for time sequencing and the common use of recommender systems algorithms in the educational data mining literature [7]. As our second baseline, we run the Bayesian Knowledge Tracing (BKT) algorithm on the data [1]. Since BKT requires a pre-defined set of concepts, we use the manually-labeled concepts that have been discovered by experts in this case.

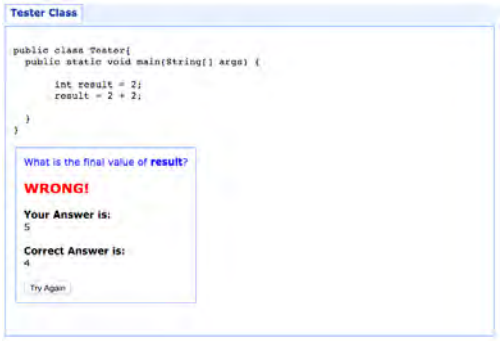


Figure 2: Screen-shot of QuizJet System

The FDTF algorithm has two parameters that need to be tuned: the number of concepts (c) and the learning rate of students (μ). We define these two parameters through cross-validation. Also, in our experiments, we set $\lambda = 0.0001$.

3.1 Dataset and Setup

We use student sequences of the QuizJet online self-assessment system to run our experiments [2]. This system produces parameterized Java quizzes based on a set of predefined templates. Hence, each student can repeat the same Java quiz, with different parameters, over and over again. The students submit their answer using a text box provided in the user interface and can receive immediate feedback. Figure 2 shows a screen-shot of this system in use.

The dataset was collected from the students who have taken a Java programming course from Fall 2010 to Spring 2013 (six semesters). The system was introduced in the class and students have voluntarily interacted with this system. The subject domain is organized by experts into 22 coherent topics. Each topic has several questions and each question is assigned to one topic. We use these sets of topics as the expert-labeled domain model in our experiments.

We experimented on 27,302 records of 166 students on 103 questions. The average number of attempts on each question is equal to three. Our dataset is imbalanced: the total number of successful attempts in the data equals 18,848 (69.04%) and the total number of failed attempts is 8454. We used a user-stratified 5-fold cross-validation to split the data so that the training set has 80% of the users (with all their records) randomly selected from the original dataset, while the remaining 20% of the users were retained for testing. In other words, 80% of students are in the training

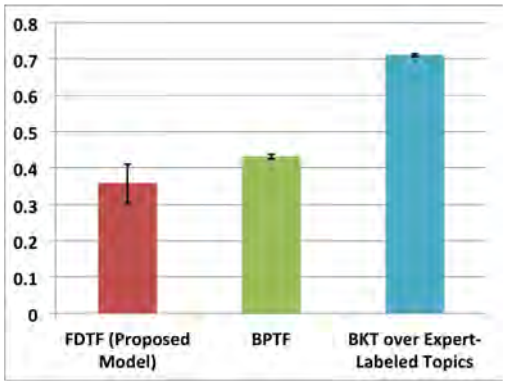


Figure 3: RMSE of Algorithms for Predicting Students Performance

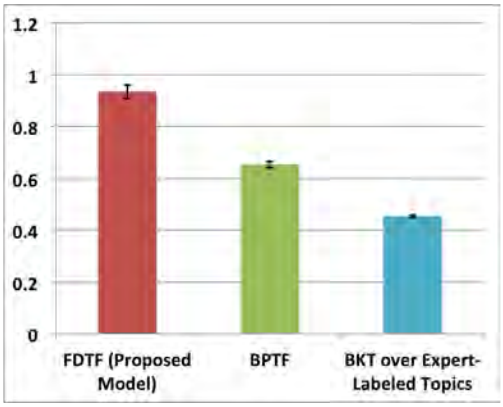


Figure 4: Accuracy of Algorithms for Predicting Students Performance

set and we have all of their sequences. For the remaining students (20%) we use 20% of their data to predict the rest 80% of it. Eventually, we include $80\% + 20\% * 20\% = 84\%$ of the whole dataset in the training set. We used the same set of data for all of the algorithms. We ran the experiments 3 times per stratification, and ended up with running each algorithm 15 times. The simple statistics of our dataset are shown in Table 1.

Table 1: Dataset Statistics

	Average	Min	Max
#attempts per sequence	3	1	50
#attempts per question	265	25	582
#attempts per student	165	2	772
#different students per question	87	7	142
#different questions per student	54	1	101

To find the best number of concepts (c) in each of the automatic PSP algorithms, we use cross-validation.

3.2 Experimental Results

As explained in Section 3, we examine the prediction performance of the proposed FDTF algorithm and the baseline models BPTF and BKT with expert-labeled topics. We then compare the accuracy of these three approaches. Since the dataset is imbalanced with approximately 70% positive labels and 30% negative labels, we define predicted values that are greater than 0.3 as positive-label predictions and predicted values that are less than or equal to 0.3 as negative-label predictions. Figure 4 shows the accuracy of the mentioned algorithms. The red, green, and cyan bars represent the accuracy of FTDF, BPTF, and BKT. As we can see in this figure, although the accuracy of the baseline tensor factorization model (BPTF) is better than Bayesian Knowledge Tracing, it is significantly less than the accuracy of the proposed approach (FDTF). Eventually, FDTF performs significantly better than both of the baseline algorithms.

Although the task of predicting student performance is a binary classification task in this setting (predicting either failure or success for students), the Root Mean Squared Error (RMSE) is traditionally used to evaluate this task in the literature. As a result, we compare the approaches based on the RMSE of approaches in addition to their accuracy. Figure 3 shows RMSE of these experiments for each of the approaches. Again, we can see that FDTF has a significantly better RMSE than both the BKT and BPTF algorithms.

These results show that, even though BKT adds the knowledge of topic-based domain model, the tensor factorization algorithms outperform it. Additionally, despite the facts that both BPTF and FDTF use the same data, model the student data as a tensor, and are temporal tensor factorization approaches, the proposed FDTF approach performs better than BPTF. These results show that explicitly modeling students' knowledge acquisition by considering their interactions with learning materials leads to better overall modeling of student knowledge, and thus provide a better overall prediction of student performance.

4. CONCLUSIONS AND FUTURE WORK

We proposed a novel tensor factorization model (FDTF) that can predict students' success or failure in future quizzes by explicitly modeling their knowledge acquisition during their interaction with learning materials. This approach does not require any expert or domain knowledge and can be automatically performed using students' historical attempt sequence. Our evaluations show that FDTF outperforms the predicting student performance approaches in the literature.

In future, we plan to explore the ability of the proposed approach in discovering the underlying domain model for the learning material, experiment on more diverse datasets, and compare our algorithm to other PSP and domain modeling approaches in the literature. We plan to improve our FDTF model to be able to model implicit feedback of students' activity, in addition to providing overall success and failure records.

The FDTF model has the potential to be used as a basis to recommend learning material to students. Also, it can help teachers discover domain models and edit or enhance learning materials, look up the concepts that students struggle to learn, and suggest appropriate learning activities.

5. ACKNOWLEDGMENTS

This work is partially supported by the Advanced Distributed Learning (ADL) Initiative (contract W911QY-13-C-0032).

6. REFERENCES

[1] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[2] I.-H. Hsiao, S. Sosnovsky, and P. Brusilovsky. Adaptive navigation support for parameterized questions in object-oriented programming. In *Learning in the Synergy of Multiple Disciplines*, pages 88–98. Springer, 2009.

[3] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review - Society for Industrial and Applied Mathematics*, 51(3):455–500, 2009.

[4] A. S. Lan, C. Studer, and R. G. Baraniuk. Quantized matrix completion for personalized learning. In *The 7th International Conference on Educational Data Mining (EDM)*, London, July 2014.

[5] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis–a new alternative to knowledge tracing. In *14th International Conference on Artificial Intelligence in Education*, volume 2009, pages 531–538, 2009.

[6] S. Sahebi, Y. Huang, and P. Brusilovsky. Parameterized exercises in java programming: using knowledge structure for performance prediction. In *The second Workshop on AI-supported Education for Computer Science (AIEDCS)*, pages 61–70. University of Pittsburgh, 2014.

[7] S. Sahebi, Y. Huang, and P. Brusilovsky. Predicting student performance in solving parameterized exercises. In *Intelligent Tutoring Systems*, pages 496–503. Springer, 2014.

[8] N. Thai-Nghe, L. Drumond, T. Horváth, A. Nanopoulos, and L. Schmidt-Thieme. Matrix and tensor factorization for predicting student performance. In *the International Conference on Computer Supported Education*, pages 69–78. Citeseer, 2011.

[9] N. Thai-Nghe, T. Horvath, and L. Schmidt-Thieme. Context-aware factorization for personalized student's task recommendation. In *Proceedings of the International Workshop on Personalization Approaches in Learning Environments*, volume 732, pages 13–18, 2011.

[10] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SIAM International Conference on Data Mining*, volume 10, pages 211–222, 2010.

Aim Low: Correlation-based Feature Selection for Model-based Reinforcement Learning

Shitian Shen
Department of Computer Science
North Carolina State University
Raleigh, NC 27695
sshen@ncsu.edu

Min Chi
Department of Computer Science
North Carolina State University
Raleigh, NC 27695
mchi@ncsu.edu

ABSTRACT

We explored a series of feature selection methods for model-based Reinforcement Learning (RL). More specifically, we explored four common correlation metrics and based on them, we proposed the fifth one named Weighed Information Gain (WIG). While much existing correlation-based feature selection methods mostly explored high correlation by default, we explored two options: *High* vs. *Low*. The former selects the next feature that has the highest correlation measure with existing selected ones while the latter selects the one with the lowest correlations. The 10 correlation-based methods were compared against previous feature selection methods for model-based RL across several datasets collected from two vastly different intelligent tutoring systems. Our results showed that the 10 correlation-based methods significantly outperform all other methods across all datasets. Among the five correlation metrics, WIG performed best. Surprisingly, for each of correlation metrics, the low option significantly outperform its high correlation peer and thus it suggests that low correlation-based feature selection methods are more effective for model-based RL than high ones.

1. INTRODUCTION

Optimal decision making in complex interactive environments is challenging. In Intelligent Tutoring Systems (ITSs), for example, system's behaviors can be treated as a sequential decision process where at each step system selects an appropriate action from a set of alternatives. Each of these system decisions will affect the user's subsequent actions and performance. Its impact on outcomes cannot be observed immediately and the effectiveness of each decision is dependent upon the effectiveness of subsequent decisions. *Pedagogical strategies* are policies that are used to decide what system action to take next in the face of alternatives.

Reinforcement Learning (RL) is one of the best machine learning approaches for decision making in interactive environments. RL focuses on inducing optimal policies on what action(s) an agent should take in any context that would maximize the agent's cumulative reward. While various RL approaches have shown promising, existing RL approaches tend to perform poorly when the interactive environment is complex in that many factors can impact desired outcomes yet not fully understood. Our general approach is to start from a collection of potentially relevant features and to apply *feature selection* methods to narrow them down to a compact and effective state representation. Many feature selection methods such as Least-squares temporal difference (LSTD) with lasso regularization [11], Monte-Carlo tree search algorithm [5] have successfully applied for RL. However, most of then are designed for model-free RL and we used model-based RL (Section 3).

In this paper, we proposed a series of correlation based feature selection methods by exploring different correlation metrics. Correlation-based methods have been widely used in supervised learning, where we use input state feature space X to predict output label Y and previous approaches mainly select the subsets of X with the *highest* correlation with the output label Y [8, 21]. However, for RL there is no output label Y and thus, to apply correlation-based feature selection methods directly to RL, we explored two options: *High* and *Low*. The former is to select the next feature that is the **most correlated (High)** with the selected ones while the latter option is to select the **least correlated (Low)** one. Theoretically speaking, choosing the most correlated feature may be effective since the selected feature is more likely to be related to decision making, however it may not make more contribution than the current selected feature set does. On the other hand, choosing the least correlated feature may raise the diversity of selected feature set and enrich the state representation, however it takes a risk of selecting irrelevant or noisy features.

In short, we explored both high and low options for five correlation metrics and resulted in 10 correlation-based methods. We compared them against an ensemble method, the methods involved in [3] referred as **RLPreviousFS** for the rest of paper, and the random feature selection method across several datasets collected from two vastly different ITSs: one is a data-driven logic tutor named Deep Thought and the other is a natural language physics tutor named Cordillera.

2. RELATED WORK

In general, existing feature selection for RL can be classified into three categories [6]: Filter, Wrapper and Embedded. Filter approaches can be seen as a preprocessing procedure in that it usually employs a ranking function so that either a fixed number of features with the highest rank or a feature set above a preset threshold value will be selected from the high-dimensional state space. This process is independent from the subsequent model learning process. For RL, the ranking function is generally based on which state feature subset would directly influence the rewards. For example, Morimoto et al. applied *kernel dimension reduction* to evaluate the conditional independence among state features and those with the most impacts on the next-time-step rewards are selected [14]. Hirotaka and Masashi [7] proposed a filter-type approach by directly evaluating the independence between immediate reward and state-feature sequences using conditional mutual information. However, it is not clear how their approach can be applied when immediate reward is not directly observable and only delayed reward is present.

Wrapper approaches search feature space and generate several candidate feature subsets, evaluate each subset using a learning algorithm, and then select the subset with the best performance. For example, Gaudel and Sebag applied Monte-Carlo tree search algorithm to generate candidate feature subsets and then evaluate the goodness of feature subset using the predefined score function [5]. In addition, Keller, et al applied LSTD to approximate value function, selected a feature subset by implementing *Neighborhood Component Analysis* to decompose approximation error, which can be used to evaluate the goodness of the feature subset [9]. Similarly, in *LSPI-FFS* Li, Williams and Balakrishnan also applied LSTD to approximate value function using linear model. They updated the parameters of the linear model through gradient descent and selected a feature subset with largest magnitude of weight [13].

Embedded approaches for RL conduct feature selection and policy induction process simultaneously. Kolter and Ng applied LSTD with *Lasso* regularization to approximate value function as well as to select effective feature subset [11]. Bach explored the penalization of approximation function by using *Multiple Kernel learning* (MLKL)[2]. Wright, Loscalzo and Yu proposed *IFSE-NEAT*, the feature selection embedded in neuroevolutionary function, which approximates the value function, and features are selected based on their contributions to the evolution of topology of network[20].

In short, while much of prior research has done on feature selection for RL, most of them is for model-free RL. For Model-based RL, Chi et al. investigated 10 filter-based methods (**RLPreviousFS**) [3]. These methods were implemented to derive a set of various policies, where features are selected mainly based on the single feature performance and the covariance in training data. Their results showed there was no consistent winner among the ten feature selection methods and in some particular cases these methods performed no better than the random baseline method. Therefore, much research on feature selection for model-based RL is needed.

3. REINFORCEMENT LEARNING & MARKOV DECISION PROCESS

Generally speaking, RL can be divided into two categories: **model-free** and **model-based**. Model-free RL [4] typically uses samples to learn a value function, from which a policy is implicitly derived. Model-based RL, by contrast, first builds up a model from samples and then compute a policy based the model. Both approaches have their own strengths and weaknesses. Model-free methods are appropriate for domains where data collection is inexpensive and trivial. Model-based methods, on the other hand, are suitable when collecting data is expensive. Given the high cost of collecting training data in our task, we focused on model-based RL and used a Markov Decision Process (MDP) framework.

MDP is defined as a tuple $\langle S, A, T, R\rangle$. S denotes state space, which reflects the generalization of interactive environment; actions A are agent's possible behaviors; reward function R can be immediate or delayed feedback from environment respect to agent's behavior and $R^a_{SS'}$ denotes the reward of transiting from state S to state S' by taking action a; transition probabilities T are defined as $T = \{p(S_j|S_i, A_k)\}^{k=1,\cdots,m}_{i,j=1,\cdots,n}$, which is estimated from training corpus. More specifically, $T^a_{SS'} = p(S'|S, a)$ denotes the probability of transiting from state S to state S' by taking action a.

Once the tuple $\langle S, A, T, R\rangle$ is set, we transform the problem of inducing effective pedagogical strategies into computing an optimal policy in an MDP by dynamic programming approaches. More specifically, we calculate the value function $V^\pi(S)$ under a policy π though Bellman equation[17], which is defined as:

$$\begin{aligned} V^\pi(S) &= E_\pi(R_t|S_t = S) \\ &= \sum_a \pi(S, a) \sum_{S'} T^a_{SS'} \left[R^a_{SS'} + \gamma V^\pi(S')\right] \end{aligned}$$

where γ is a constant called discount factor. The optimal value function can be estimated by

$$V^*(S) = \max_\pi V^\pi(S)$$

Then we can derive the optimal policy corresponding to the optimal value function $V^*(S)$. Here we used the toolkit developed by Tetreault and Litman [18]. Besides inducing an optimal policy, Tetreault, & Litman's toolkit also calculate the Expected Cumulative Reward (ECR) for the induced policy. The ECR of a policy is derived from a side calculation in the policy iteration algorithm: the V-values of each state, the expected reward of starting from that state and finishing at one of the final states. More specifically, the ECR of a policy π can be calculated as follows:

$$ECR_\pi = \sum_{i=1}^{n} \frac{N_i}{N_1 + \cdots + N_n} \times V(s_i) \qquad (1)$$

Where $s_1, \cdots, s_n$ is the set of all starting states and $V(s_i)$ is the V-values for state s_i; N_i is the number of times that s_i appears as a start state in the model and it is normalized by dividing $\frac{N_i}{N_1+\cdots+N_n}$. In other words, the ECR of a policy π is calculated by summing over all the initial start states in the

space and weighting them by the frequency with which each state appears as a start state. The higher the ECR value of a policy, the better the policy is supposed to perform.

In our application, we defined our action set A and reward function R in Section 5. However the state space S is not well-defined, where each state is a vector representation composed of a fixed number of state features $F = \{F_1, F_2, ..., F_p\}$. Our approach is to apply various feature selection methods to narrow a wide set of feature space to a compact and effective subset that would model student learning process accurately.

4. METHODOLOGY

In this section, we first describe the five basic correlation metrics we used and then describe our general feature selection procedure. More specifically, we will describe our 10 correlation-based methods, the ensemble method, and finally briefly describe the RLPreviousFS methods.

4.1 Five Correlation Metrics

In order to quantize correlation among features, we used five correlation metrics. The first four are commonly used in supervised learning and here we will investigate whether they can be applied to RL. We proposed the fifth one, Weighted Information gain, by combining the four commonly used metrics and adapting them based on the characteristic our task and datasets. More specifically, we have:

1. Chi-squared (CHI)[22]: a statistical test used to identify whether the distribution of a categorical variable differ from the other one, which induces the independence between two variables. CHI is usually applied to evaluate the independence of two variables in mathematical statistics.

2. Information gain (IG)[12]: it measures the differ between the uncertainty of a variable Y and the uncertainty of Y given variable X as conditional information. It is calculated as:

$$IG(Y, X) = H(Y) - H(Y|X)$$

where $H()$ is called entropy function, measure uncertainty of a variable. IG evaluates the certainty of variable Y obtained from variable X, which can be treated as one type of correlation between X and Y. IG has the bias towards the variable with a large number of distinct values.

3. Symmetrical certainty (SU)[21]: it is defined as:

$$SU(Y, X) = \frac{H(Y) - H(Y|X)}{H(X) + H(Y)}$$

SU evaluates the correlation between two variables by normalizing IG. SU compensates the weakness of IG and it is a symmetrical measurement, which treats a pair of variables symmetrically.

4. Information gain ratio (IGR)[10]: it's the ratio of information gain to the intrinsic information, which is the entropy of conditional information. IGR can be represented as:

$$IGR(Y, X) = \frac{H(Y) - H(Y|X)}{H(X)}$$

Comparing with IG, IGR takes the uncertainty of conditional information into account with purpose of removing bias of selecting variable with many distinct values. However, IGR is not a symmetrical measurement ($IGR(X, Y) \neq IGR(Y, X)$).

5. Weighted Information gain (WIG): it is proposed as:

$$WIG(Y, X) = \frac{H(Y) - H(Y|X)}{(H(Y) + H(X))H(X)}$$

We propose WIG by combining IG, SU and IGR. Comparing with IGR, WIG normalized IG by considering the uncertainty of both variables X and Y and also compensate the weakness of IG. Comparing with SU, although WIG is not symmetrical measurement. Based on the above equation, WIG sets more weight for variable X. In our application, WIG is used for evaluating the correlation between current selected feature set Y with the new feature X .

For each of the five correlation metrics, we explored two options: High and Low, which resulted in 10 correlation-based methods named five High methods: CHI-high, IG-high, SU-high, IGR-high, WIG-high and five Low methods: CHI-low, IG-low, SU-low, IGR-low, and WIG-low. Our goal is to investigate which option is better: high vs. low and which of the five correlation metric performs the best.

4.2 Correlation-based Feature Selection

In this project, we followed a forward stepwise feature selection procedure in that: given current selected feature set, our correlation-based methods select the feature forwardly based on the five correlation metrics described above.

Algorithm 1 Correlation-based Feature Selection Algorithm

Require: Ω: Feature space; $\mathcal{D}$: Training data; $\mathcal{N}$: Maximum number of selected features
Ensure: $\mathcal{S}^*$: Optimal feature set
for f_i in Ω **do**
  $ECR_i \leftarrow$ CALCULATE-ECR($\mathcal{D}$, f_i)
end for
Add f^* with highest ECR to $\mathcal{S}^*$
while SIZE($\mathcal{S}^*$) $< \mathcal{N}$ **do**
  for f_i in $\Omega - \mathcal{S}^*$ **do**
    $C_i \leftarrow$ CALCULATE-CORRELATION($\mathcal{S}^*$, f_i, m)
  end for
  $\mathcal{F} \leftarrow$ SELECTTOP(C, 5, reverse) ▷ Select top 5 features based on correlation metrics
  for f_i in $\mathcal{F}$ **do**
    $ECR_i \leftarrow$ CALCULATE-ECR($\mathcal{D}$, $\mathcal{S}^* + f_i$)
  end for
  Replace $\mathcal{S}^*$ by $\mathcal{S}^* + f_i$ with higgest ECR
end while

Algorithm 1 shows the concrete process of our correlation-based feature selection procedure. It contains three major

parts: in the first part (lines 1–4), it constructs MDPs for each single feature, induces a single-feature policy and calculates it ECR. Then the feature with highest ECR is added into current optimal feature set. In the second part (lines 6–9), it evaluates the correlations between current optimal feature set $\mathcal{S}^*$ with other features $f_i \in \Omega - \mathcal{S}^*$, ranks the correlations, and then selects the top 5 highest ones for high correlations or the bottom 5 lowest ones for low correlations. They are selected to form a feature pool $\mathcal{F}$. In the third part (lines 10–13), several candidate feature sets are generated by combining current optimal feature set $\mathcal{S}^*$ with each feature $f_i \in \mathcal{F}$. Then ECR for each candidate feature set can be evaluated by applying *Calculate-ECR* function. Current optimal feature set $\mathcal{S}^*$ will be replaced by the candidate feature set with highest ECR. The algorithm will terminate until the size of optimal feature set reaches maximum number $\mathcal{N}$. The third part can be treated as the process of wrapper approach where several candidate feature sets are evaluated by the RL method. Therefore, our correlation-based methods are the combination of filter and wrapper approaches.

4.3 Ensemble Method

Our ensemble approach combines the 10 proposed correlation-based methods and 4 *RL-based* methods (Section 4.4), which are most effective methods among RLPreviousFS. Its procedure is similar to that of correlation-based method except the second part (lines 6–9). The ensemble approach integrates the features generated from each method and generates a relatively big feature pool $\mathcal{F}$. The maximum size of $\mathcal{F}$ is up to 70 but often smaller because of the overlapping feature sets. Note that it is still much larger than any of our 10 correlation-based methods which has 5 candidates for each step. After generating the feature pool, the ensemble method jumps to the third part (lines 10-13) of Algorithm 1. At each step, the ensemble method explores feature sets by adding the feature with maximum ECR.

4.4 RLPreviousFS

Chi et. al [3] grouped RLPreviousFS into three categories:1) four RL-based methods; 2) two PCA-based method, which selects features with the high correlation with principle components; 3) four PCA&RL-based methods, which use RL-based methods to select features from a candidate feature set which is generated from PCA-based method. All three categories can be seen as the filter approaches.

5. TRAINING DATASETS

5.1 Two Deep Thought Datasets

Deep Thought (DT)[15] is a data-driven ITS . It is a rule-based system where students need to select different rules to complete logic proof problems. In DT, we focused on a problem level decision named problem solving (PS) vs. Worked Example(WE). More specifically, when starting the next training problem, the tutor will make a simple decision: "should it ask student to solve the next problem **(PS)**, or should it provide an example to show the student how to solve the next problem **(WE)**".

Our training dataset includes a total of 303 undergraduate CS students who used DT as part of class assignment in Fall 2014 and Spring 2015. The average amount of time spent in the tutor was 416.60 minutes. To induce RL policies, a total of 134 features were extracted from the student-system log files. The reward function in DT dataset is calculated based on level score $LevelScore_i$ where $i \in [1, 6]$. Particularly, we designed two type of reward: immediate and delay reward. Immediate reward is defined as $R_i = LevelScore_i - LevelScore_{i-1}$ where $i \in [1, 6], R_1 = LevelScore_1$, it reflects the change of students' performance level by level. Delayed reward is represented as $R_{delay} = LevelScore_6 - LevelScore_1$, which determines the change of students' performance across all levels. For the convenience, we denote the two DT datasets with immediate reward as *DT-Immed* and that with delayed reward as *DT-Delay* respectively.

5.2 Six Cordillera Datasets

Cordillera [19] is a natural language tutoring system teaching college introductory physics. Different from DT tutor system, Cordillera requires students to input their answer by natural language free text. The data collection consists of the following stages: 1) background survey; 2) studying textbook and prerequisite materials, 3) taking a pretest; 3) training on Cordillera, 4) and taking a post test. Cordillera makes step-level decision: *Elicit/Tell* **(ET)**. The ET decision means "should the tutor system *elicit* the next problem-solving step for student, or should it *tell* student the instruction of next step directly".

Our training corpus involves 64 students. In Cordillera, there are five primary Knowledge Components (KCs): Definition of Kinetic Energy (KE), Gravitational Potential Energy (GPE), Spring Potential Energy (PE), Total Mechanical Energy (TME), and finally Conservation of Total Mechanical Energy (CTME). In STEM domains such as math and science, it is commonly assumed that the relevant knowledge is structured as a set of independent but co-occurring KCs. A KC is "a generalization of everyday terms like concept, principle, fact, or skill, and cognitive science terms like schema, production rule, misconception, or facet" [19]. For the purposes of ITSs, these are the atomic units of knowledge. It is assumed that a tutorial dialogue about one KC (e.g., kinetic energy) will have no impact on the student's understanding of any other KC (e.g, of gravity). This is an idealization, but it has served ITS developers well for many decades, and is a fundamental assumption of many cognitive models [1, 16]. Given the KCs' independence assumptions, we will apply RL to induce KC-specific pedagogical strategies for each of the five primary KCs individually. Moreover some steps in Cordillera have mixed KC, thus we also apply RL to induce pedagogical policies irregardless of the KCs involved (denoted by Across). In short, we have a total of **six** Cordillera KC datasets, one per KC for the five primary KCs and one KC-general for the Across policy. Each of the KC datasets contains 50 state features and to induce RL-rules, we used the delayed reward defined as student Normalized Learning Gains (NLGs): $NLG = \frac{Posttest - Pretest}{MaximumScore - Pretest}$. Here $MaximumScore$ is the maximum score a student can get and for both pretest and posttest, the maximum score is set to be 1.

6. EXPERIMENT & RESULT

To evaluate the effectiveness of induced policies, we set the maximum number of selected features to be 6 considering the size of our training datasets. In this section, we present

Table 1: The highest ECR Induced by Correlation-based Methods Across Eight Datasets

ITS	Data	CHI		IG		SU		IGR		WIG	
		High	Low	High	Low	High	Low	High	Low	High	Low
DT	Immed	55.89	129.82	53.87	95.81	53.87	95.81	53.87	95.81	59.04	**143.16***
	Delay	8.89	12.56	8.89	12.58	10.73	12.58	8.94	**15.43***	8.94	**15.43***
Cordillera	KE	5.86	6.75	5.86	6.75	5.86	6.75	5.57	**7.64***	5.57	7.62
	GPE	10.47	13.39	11.80	13.39	11.21	13.39	11.10	**17.23***	10.82	**17.23***
	SPE	12.67	17.17	12.67	14.88	12.67	**18.02***	10.83	**18.02***	10.27	**18.02***
	TME	7.34	7.96	7.57	9.42	7.47	9.42	6.98	**10.04***	6.40	**10.04***
	CTME	23.01	32.71	24.01	31.22	24.31	31.22	23.01	**33.24***	23.01	**33.24***
	Across	1.77	2.26	1.77	2.26	1.77	**2.57***	1.77	2.26	1.77	**2.57***

Note: The best ECR among 10 methods for each dataset is highlighted by *.

Table 2: Overall Evaluation Across Eight Datasets

	DT		Cordillera					
	Immed	Delayed	KE	GPE	SPE	TME	CTME	Across
Low Correlation	**143.16**	**15.43**	**7.64**	**17.23**	**18.02**	**10.04**	**33.24**	2.57
High Correlation	59.03	10.72	5.85	11.80	12.67	7.57	24.31	1.71
Ensemble	127.79	12.61	7.33	16.40	16.95	9.12	32.06	**2.68**
RLPreviousFS	60.28	12.56	6.17	14.41	11.90	7.15	24.60	2.03
Random	8.53	7.62	4.26	7.34	10.52	4.78	22.02	1.20

the experimental analysis of the correlation-based methods, the ensemble, the RLPreviousFS used in previous research, and random feature selection methods which is our baseline method.

6.1 Comparing correlation-based methods

In this section, we want to answer two questions:
1) which option is better for model-based RL: High vs. Low;
2) which of the five correlation metrics performs the best.

High VS Low. Table 1 shows the performance of the 10 correlation based methods across eight training datasets: two DT and six Cordillera datasets. The rows represent the eight datasets while columns represent the 10 correlation-based methods. Each cell in Table 1 shows the highest ECR of the policy generated from the corresponding correlation-based feature selection method on the corresponding dataset when the number of features varies from 1 to 6.

Table 1 shows that for each of five correlation metrics, the low correlation-based method significantly outperform its high correlation-based peer. For *DT-Immed* dataset, the ECR of WIG-low is 143.16, while ECR of WIG-High is only 59.04; the former is 140% higher than the latter. Similarly, the $ECRs$ of CHI-low and CHI-High are: 129.82 vs. 55.89 and the former is 132% higher than the latter. The similar results is true across all five correlation metrics and across all eight datasets.

Moreover, the out-performance of the Low option over the High option seems to be more prominent on DT datasets than Cordillera datasets. For DT data, the average percent increase for the low correlation methods over the high correlation methods is 75.35%, the maximum percent increase is 142.48% and the minimum percent increase is 17.24%. For Cordillera KC datasets, the average percent increase for the low correlation methods over the high ones is 35.15%, the maximum percent increase is 75.46% and the minimum percent increase is 8.45%. On average the low correlation methods outperform the high correlation peers by 45.2%.

To summarize, our results showed that the low correlation option is more suitable for the model-based RL than the high correlation option. It indicates that it is important to include a variety of features in the state representation for applying RL to induce pedagogical policies.

Five Correlation Metrics. In Table 1, for each of the eight datasets, we highlight the best ECR of the induced policies by *. Table 1 shows that the WIG is the consistent winner in that it has the best ECR for all datasets except for *KE*. On the KE dataset, WIG-Low performance is slightly lower than the best policy: 7.62 for WIG-Low vs. the highest 7.64 for IGR-Low. Following WIG, IGR is the second best in that it has the highest ECR for six out of eight datasets. Note that WIG and IGR together produced all the best policies across all eight datasets and they overlapped on *DT-Delay, GPE, SPE, TME, CTME*. Except for WIG and IGR, the remaining three metrics only induced 2 best policies and both are found by SU-Low. In short, our proposed WIG performed the best among the five correlation metrics followed by IGR.

6.2 Overall Evaluation

Table 2 shows the overall comparison among all feature selection methods. With the purpose of simplicity, for the five low-correlation methods, the five high-correlation methods and the RLPrevousFS methods, we select the best one from each category. Thus, Table 2 will compare the five categories of feature selection methods: the best of the five Low-

correlations, the best of the five High-correlations, the ensemble, the best of RLPreviousFS and the random method.

In Table 2, rows denote the five categories and columns show the eight datasets. Table 2 shows that as expected the random method performs the worst across all datasets. In addition, the best of the low correlation-based methods outperforms all other methods in all datasets except in the *Across* dataset, where the ensemble method performs slightly better that the best of the low correlation-based methods. On average, the best low correlation-based method increases over the best of RLPreviousFS by 43.87% and over the ensemble method by 9.05%. In addition, the ensemble method improves over the best of RLPreviousFS on average 36.46%. To summarize, we can rank the five categories of methods as Low correlation-based > Ensemble > High correlation-based, RLPreviousFS $\gg$ Random.

7. CONCLUSIONS & FUTURE WORK

In this paper, we proposed 10 correlation-based feature selection methods for model-based RL. Our result clearly showed that the low correlation-based methods are more effective than the ensemble, the high correlation-based, the RLPreviousFS, and the random method. Among the five correlation-based metrics, our proposed WIG performed the best. WIG found the best policies across all eight datasets except that on KE, its performance is only slightly lower than the best one which is found by IGR.

While in supervised learning features associated with highest correlation are generally selected, for model-based RL selecting the next feature with lowest correlation is more effective. Moreover, it is surprising to see that the ensemble method only performed the best on one out of eight datasets. Given that the motivation for applying the ensemble method is that it can take the advantages of each method with purpose of achieving better results. Therefore, one of our future work is to explore other ways to make our ensemble method more effective.

8. ACKNOWLEDGEMENTS

This research was supported by the NSF Grant 1432156 "Educational Data Mining for Individualized Instruction in STEM Learning Environments".

9. REFERENCES

[1] J. R. Anderson. *The architecture of cognition*. Cambridge, Mass. : Harvard University Press, 1983.

[2] F. R. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *NIPS*, pages 105–112, 2009.

[3] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction*, 2011.

[4] N. D. Daw. Model-based reinforcement learning as cognitive search: neurocomputational theories. *Cognitive search: Evolution, algorithms and the brain*, pages 195–208, 2012.

[5] R. Gaudel and M. Sebag. Feature selection as a one-player game. In *ICML*, pages 359–366, 2010.

[6] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[7] H. Hachiya and M. Sugiyama. Feature selection for reinforcement learning: Evaluating implicit state-reward dependency via conditional mutual information. In *Machine Learning and Knowledge Discovery in Databases*, pages 474–489. Springer, 2010.

[8] M. A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[9] P. W. Keller, S. Mannor, and D. Precup. Automatic basis function construction for approximate dynamic programming and reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 449–456. ACM, 2006.

[10] J. T. Kent. Information gain and a general measure of correlation. *Biometrika*, 70(1):163–173, 1983.

[11] J. Z. Kolter and A. Y. Ng. Regularization and feature selection in least-squares temporal difference learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 521–528. ACM, 2009.

[12] C. Lee and G. G. Lee. Information gain and divergence-based feature selection for machine learning-based text categorization. *Information processing & management*, 42(1):155–165, 2006.

[13] L. Li, J. D. Williams, and S. Balakrishnan. Reinforcement learning for dialog management using least-squares policy iteration and fast feature selection. In *INTERSPEECH*, pages 2475–2478, 2009.

[14] J. Morimoto, S.-H. Hyon, C. G. Atkeson, and G. Cheng. Low-dimensional feature extraction for humanoid locomotion using kernel dimension reduction. In *ICRA*, pages 2711–2716. IEEE, 2008.

[15] B. Mostafavi, G. Zhou, C. Lynch, M. Chi, and T. Barnes. Data-driven worked examples improve retention and completion in a logic tutor. In *Artificial Intelligence in Education*, pages 726–729. Springer, 2015.

[16] A. Newell. *Unified Theories of Cognition*. Harvard University Press; Reprint edition, 1994.

[17] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

[18] J. R. Tetreault and D. J. Litman. A reinforcement learning approach to evaluating state representations in spoken dialogue systems. *Speech Communication*, 50(8):683–696, 2008.

[19] K. VanLehn, P. W. Jordan, and D. J. Litman. Developing pedagogically effective tutorial dialogue tactics: experiments and a testbed. In *SLaTE*. Citeseer, 2007.

[20] R. Wright, S. Loscalzo, and L. Yu. Embedded incremental feature selection for reinforcement learning. Technical report, DTIC Document, 2012.

[21] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, volume 3, pages 856–863, 2003.

[22] M. F. Zibran. Chi-squared test of independence. *Department of Computer Science, University of Calgary, Alberta, Canada*, 2007.

Personalization of Learning Paths in Online Communities of Creators

Mingxuan Sun*
Division of Computer Science and Engineering
Louisiana State University
msun@csc.lsu.edu

Seungwon Yang
School of Library and Information Science
Center for Computation and Technology
Louisiana State University
seungwonyang@lsu.edu

ABSTRACT

In massive online communities of creators (OCOCs), one of the core challenges is to encourage users to learn to create original contents using basic components. Recommending the right learning components at the right time is critical for improving user engagement and has not been fully studied due to the unstructured nature of online communities. To address the problem, we propose in this paper a novel recommendation model which integrates Cox's survival analysis and collaborative filtering. Our model can incorporate factors such as user learning history and social engagements, which provides us insights in improving the personalized service. We apply our method to the user data from Scratch online platform and demonstrate the performance of the model.

1. INTRODUCTION

In recent years, the number of online learning communities (OCOCs) has increased exponentially as evidenced by successful platforms such as Scratch online[1]. These online communities offer flexible learning environment where users can create projects (e.g., games, art designs), share projects, and engage with like-minded users in the community. One of the goals is to foster learning programming concepts through developing and sharing projects among its users based on interactions in the community [11]. Previous studies [7] have found that creating and sharing projects is the gateway to other online social activities including commenting and following. However, only about 29% of Scratch users would like to share their projects and about half of them contribute no more than one project.

One way to improve user engagement is to track users' learning history and recommend contents tailored to each individual. For example, Scratch users learn to create projects by manipulating basic programming blocks such as "goto", "changecolor", and "doIf". Each block is categorized in a certain Computational Thinking (CT) concept [6]. Users are expected to learn CT concepts such as "motion" by manipulating blocks such as "goto", "bounce", and "turn". Users may follow different learning paths over time. Based on programming blocks that each user has used in his/her previous projects, we can recommend particular blocks, concepts, or projects tailored to the individual. For instance, for users who are interested in animation projects with some basic motion blocks such as "goto", the system can recommend projects that have more advanced motion blocks such as "bounce".

In addition to what to recommend, when is a good time to recommend is another important factor to consider since suggesting blocks to users at the right time may influence learning effectiveness and efficiency. For example, if a user is still struggling with basic motion techniques such as "goto", it may not be a good idea to introduce a project or a more advanced programming concept such as "turn" or "direction". Our goal is to alleviate the high dropout rates in the early stage through personalization of the learning path.

In this paper, we propose a model to learn the probability of a user's exposure to a certain learning component at a particular time. The probability of exposure is estimated based on a collaborative filtering model, which recommends the user the items favored by the like-minded. The conditional probability of a user being exposed to a given item at a particular time is modeled by the Cox proportional hazard model from survival analysis.

2. RELATED WORK

Early studies on learning behavior analysis for OCOCs have been based on case-studies evaluating learning process qualitatively [5, 12]. Other attempts [3, 7] have focused on clustering user behaviors based on types and volumes of users' online activities. A recent work by Yang et al. [14] modeled *informal learning trajectories* quantitatively as the growth of cumulative usage of programming blocks by each user.

Personalization approaches that are based on user behaviors have been widely studied in different types of Web services such as e-commerce. In e-commerce, most personalization approaches focus on recommending users the items that have been favored by like-minded users based on their purchase history. Traditional recommendation algorithms are memory based methods including vector similarity and correla-

*Corresponding author.
[1]https://scratch.mit.edu/

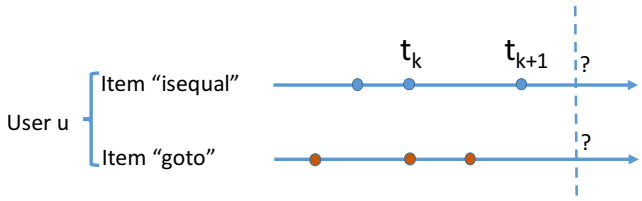


Figure 1: Time-aware recommendation. The occurrence time of user-item interaction is modeled using survival analysis. Our goal is to predict the most desired learning item i at a particular time t for each individual user u.

tion [2]. The state-of-the-art methods including the one that won the Netflix competition [9] are based on matrix factorization. The time factor in personalization services largely affects the user satisfaction of the service [13, 10]. Our contribution in this paper lies in that we incorporate both the Cox model and collaborative filtering to provide personalized recommendation for online learners.

3. METHOD

In OCOCs, users create and share projects consisting of basic items such as programming blocks in Scratch. Each item belongs to a certain category. Based on user-item interaction histories, we would like to suggest items tailored to each user at a particular time. To achieve this goal, we propose to estimate the joint probability $p(u, i, t) = p(t|u, i)p(u, i)$, where $p(u, i)$ is the probability of user u interacting with item i and $p(t|u, i)$ is the conditional probability of user u interacting with item i at time t.

We model the occurrence time t of the event that user u interacts with item i using the Cox model in survival analysis. Survival analysis is used to estimate the probability of the occurrence of an event $p(\text{event in } [t, t + \Delta t])$ such as when a patient fails to survive. In the online learning context, our task is to estimate the probability of the occurrence of exposing to a specific learning block for each user, which is $p(t|(u, i))$. As shown in Figure 1, in the observed sequences of user-item interactions, a user builds a project with a set of items (e.g., "isequal" and "goto") at time t_k. Then item i is used again in another project of the same user at time t_{k+1}. Let x_k be the covariates associated with user u at time t_k. We are interested in predicting the time gap $t_{k+1} - t_k$.

Let $\lambda(t)$ denote the instantaneous rate of event happening at time t following the last event given the covariates x_k, that is $\lambda(t) = P(T = t \mid T \geq t)$. The Cox model assumes that the covariates only affect the magnitude of each individual hazard rates. Formally, for an individual observation with covariates x_k, the hazard at time t is:

$$\lambda(t) = \lambda_0(t) * \exp(x_k^T \beta), \tag{1}$$

where λ_0 is the non-parametric baseline hazard function, x_k is the covariates, and β is the regression coefficient. The log likelihood of observing the occurrences is:

$$\log L = \sum_{k=1}^{K} \left\{ d_k \log \lambda(t_k) - \int_0^{t_k} \lambda(\tau) d\tau \right\}, \tag{2}$$

where d_k is a censor indicator, taking the value one if event occurs at time t_k or the value zero if event does not occur till time t by the end of observation window. The parameters β and the baseline hazard λ_0 can be estimated by maximizing the log partial likelihood with Breslow's approximation [4].

We further estimate the probability $p(u, i)$ of a user favoring a particular item (e.g., block) by adopting collaborative filtering (CF) recommendation algorithms. User interactions contain substantial information to improve recommendation accuracy. For example, in Scratch, users play with a set of programming blocks to develop a project. Therefore, the frequency of each type of block may indicate their preferences. Based on the previous learning history, the system can predict interesting blocks tailored to individual taste. Collaborative filtering methods focus on detecting users with similar preferences and recommending items favored by the like-minded. Algorithms range from similarity based CF methods [2] to matrix factorization based CF methods popularized by the Netflix Prize Competition [9].

Let r_{ui} denote the observed preference of user u for item i, where $u = 1, 2, \ldots, m$ and $i = 1, 2, \ldots, n$. The pairs (u, i) are stored in the set $O = \{(u, i) \mid r_{ui} \text{ is observed}\}$. Since the observed ratings or event frequencies are very sparse, matrix factorization is used to learn latent features of both users and items in a lower dimensional space such that the product of each user-item pair can best approximate the ratings. Specifically, let θ_u and v_i denote latent features for user u and items i, where θ_u and v_i are k-dimensional vectors. The latent features can be estimated by minimizing a prediction loss function between the predicted ratings and true ratings of users. That is,

$$\min_{\Theta, V} \sum_{(u,i) \in O} (r_{ui} - \theta_u^\top v_i)^2, \tag{3}$$

where $\Theta = [\theta_1, \theta_2, \ldots, \theta_m]$ is a $k \times m$ matrix and $V = [v_1, v_2, \ldots, v_n]$ is a $k \times n$ matrix. A gradient descent based method [9] can be used to estimate latent features. The probability of user favoring an item $p(u, i)$ can be generated using a softmax function:

$$p(u, i) = \frac{\exp(r_{ui})}{\sum_{j=1}^{n} \exp(r_{uj})}, \tag{4}$$

4. EXPERIMENTAL RESULTS

We evaluate the model performance through two steps: time-to-return prediction and time-aware recommendation. In the first step, for every user-item interaction (u, i), we estimate the probability of the next occurrence at time t and use the expected value of the time as the predicted time to return. In the second step, for each user u at a particular time t, we rank each item i by the joint probability $p(u, i, t)$ and recommend top-K items. We present the experimental details including data collection, evaluation metrics, and competing baselines.

4.1 Data Collection

We apply our method to user data which was released in spring of 2014 from Scratch online[2]. Users can create a project by programming with basic components called blocks. Each block can be categorized into one or more CT concepts.

[2]https://llk.media.mit.edu/scratch-data/

Table 1: Covariate analysis for CT concept "conditionals". *** p<0.001, ** p<0.01, * p<0.05, . p<0.1

Covariate Name	Coefficient	P-Value
is.remix	0.190556	0.000593 ***
is.self.remix	-0.140119	0.062772 .
is.remixed	0.447668	2.44e-15 ***
like 2 or more	0.226432	0.001440 **
follow 2 or more	0.262599	0.000346 ***
comments 2 or more	0.478668	< 2e-16 ***
conditionals experience	0.332191	1.14e-08 ***
operators experience	-0.074161	0.236036
data experience	-0.157914	0.010259 *

Table 2: Covariate analysis for CT concept "data". *** p<0.001, ** p<0.01, * p<0.05, . p<0.1

Covariate Name	Coefficient	P-Value
is.remix	0.15007	0.018629 *
is.self.remix	-0.18239	0.037235 *
is.remixed	0.52571	3.33e-16 ***
like 2 or more	0.23287	0.005221 **
follow 2 or more	0.20475	0.023510 *
comment 2 or more	0.54465	< 2e-16 ***
conditionals experience	0.03648	0.605257
operators experience	0.14616	0.041800 *
data experience	0.12105	0.076548 .

We adopt the the mapping table from blocks to CT concepts as suggested in [6]. Users are encouraged to share their projects and interact with others by commenting projects, favoring projects, or following other users. For each user, the dataset includes the project details including block usage and timestamps. It also maintains tables of different types of social interactions including user follower-followed relationship and comments. The user history data collected from December 2011 to March 2012 are used to create the training and the testing datasets. Possible spam users who create more than 100 projects in a day are filtered out. The remaining data contains 22415 users and 170 learning blocks with 6 CT concepts. All user records observed during December 2011 to February 2012 are used to train the model through cross-validation and all user records during March 2012 are used for testing.

The following covariates are used to estimate the Cox model. Covariates related to user activity history include the number of days since registration and the gap since last login. User social interaction covariates include the number of projects liked, the number of friends followed, and the number of comments on projects. User project details include the number of projects created, the number of types of blocks, and the number of concepts. We collect user covariates on a daily basis and predict the days till the user's next event. Users who had not been exposed to the event by the end of the time window were censored.

4.2 Performance Evaluation

In the first step "time-to-return prediction", for every block pair (u, i), we estimate the probability of the next occurrence at time t and treat the expected value of the time as the predicted time to return. Since the data are sparse, a direct estimation of a survival model for each block will be noisy. Instead, we train a Cox model for each CT concept using the interactions events of blocks belonging to that concept. To evaluate the performance, we predict the expected time from the learned density function and compute the Rooted Mean Square Error (RMSE) with respect to the true time. We compare the Cox model against the baselines including linear regression and decision tree regression. Smaller RMSE values indicate better performance.

The importance of covariates for predicting each individual user's exposure to CT concepts "conditionals" and "data" are shown in Tables 1 and 2. Both tables show the covariates' names, the regression coefficients and the significance scores. A positive regression coefficient for a variable implies a higher hazard if the value of the variable is high. Both tables show that the regression coefficients for the variable "is.remixed.bool" are positive. It indicates that if a user's project is remixed by others, the hazard rate of observing the user's next event will increase by a factor of $exp(0.190556) - 1$ compared with the baseline hazard. On the contrary, a negative regression coefficient implies a lower hazard, which means the probability of user interacting with the blocks belonging to that concept will be smaller. The value of the coefficient is statistically significant at different significance levels. We only show the covariates with highest significant levels.

As shown in the tables, for both CT concepts "conditionals" and "data", for users who share projects later remixed by others, it is more likely that these users will be back creating projects in the future. Interestingly, users who remix others' projects will be more likely to create projects than those who remix their own projects. In addition, users who like two or more projects, who follow two or more friends, and who have two or more comments are more likely to create and share projects in the future than those who have no social interactions. This implies that social interactions help users to learn and share. In addition, we can see that users who have built blocks in the concept "conditional" are more likely to build blocks falling into the same concept. Interestingly, users who have built blocks in the concepts "operator" and "data" are more likely to build blocks in the concept "data".

We then use the estimated model to predict the time to the next event in each CT concept. Table 3 displays the root mean square error (RMSE) for the return time prediction using the Cox model and baselines, respectively. For concepts "loops", "conditionals", "operators", and "data", the hazard based approach outperforms all the other baselines. For concept "event", the hazard based approach performs very close to linear regression and both of them perform better than the others. All the baselines do not model the underlying temporal patterns in the observed sequences.

For the final step "time-aware recommendation", suppose the testing event of user u occurs at time t, we compute the probability $p(u, i, t)$ of the user favoring an item i at time t for each item i and rank among all items by probability. Ideally, the observed items that the user actually interacts with should appear on top positions. In information retrieval, we focus on the evaluation accuracy on top positions using several standard metrics including precision at k (P@k), Mean Average Precision (MAP) and Normalized Discounted

Table 3: RMSE comparison for user return time prediction. Smaller values indicate better performance.

	Loops	Events	Conditionals	Operators	Data
Linear Regression	9.13	**9.20**	8.94	8.79	8.68
Decision Tree Regression	9.33	9.41	9.13	9.00	8.80
Cox model	**9.04**	9.25	**8.63**	**7.97**	**7.62**

Table 4: Comparison of recommendation accuracy.

	P@1	P@3	P@5	MAP@20	NDCG@20
NMF	0.78	0.70	0.64	0.71	0.67
SurvMF	0.84	0.72	0.64	0.72	0.68

Cumulative Gain at k(NDCG) [8]. We compare with the state-of-the-art baseline non-negative matrix factorization (NMF) [1]. We follow the standard procedure in collaborative filtering to estimate the model using the user data in the training set and evaluate the performance of the prediction in the test set. Specifically, the user records observed before March 2012 are used to train and the user records in March 2012 are used to test. The data contains the rating of each user-block pair, where the rating corresponds to the categorization of event occurrences. The maximum rating is 6 for six or more event occurrences. At the time of the testing event, we compare the ranked list with ground truth. As shown in Table 4, since our method (SurvMF) integrates the survival model into the matrix factorization to capture the temporal dynamics of user-item interaction, it can achieve better performance.

5. CONCLUSIONS AND FUTURE WORK

In this work, we have focused on personalization of learning path in massive online communities of creators. One of the main challenges in online learning is high dropout rates in the early stage due to cognitive overload. To alleviate the problem, we propose a novel model integrating the Cox model and matrix factorization to recommend the right learning contents at the right time. The model can incorporate factors such as user learning history and social engagements. In addition, the latent features learned through matrix factorization further improves the recommendation accuracy. Empirical evaluations on real world data demonstrate the performance of our model.

References

[1] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational statistics & data analysis*, 52(1):155–173, 2007.

[2] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998.

[3] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 280–284. ACM, 2000.

[4] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.

[5] A. Dahotre, Y. Zhang, and C. Scaffidi. A qualitative study of animation programming in the wild. In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '10, pages 29:1–29:10, New York, NY, USA, 2010. ACM.

[6] S. Dasgupta, W. Hale, A. Monroy-Hernández, and B. M. Hill. Remixing as a pathway to computational thinking. In *ACM Conference on Computer-Supported Cooperative Work and Social Computing*, pages 1438–1449. ACM Press, 2016.

[7] D. Fields, M. Giang, and Y. Kafai. Understanding collaborative practices in the scratch online community: Patterns of participation among youth designers. *To see the world and a grain of sand: Learning across levels of space, time, and scale: CSCL 2013 Conference Proceedings*, 2013.

[8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[9] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data*, 4(1):1–24, 2010.

[10] J. Lehmann, M. Lalmas, E. Yom-Tov, and G. Dupret. Models of user engagement. In *User Modeling, Adaptation, and Personalization*, pages 164–175. Springer, 2012.

[11] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, and Y. K. B. Silverman. Scratch: programming for all. *Communications of the ACM*, 52(11):60–67, 2009.

[12] C. Scaffidi and C. Chambers. Skill progression demonstrated by users in the scratch animation environment. *Int. J. Hum. Comput. Interaction*, 28(6):383–398, 2012.

[13] J. Wang and Y. Zhang. Opportunity model for e-commerce recommendation: right product; right time. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 303–312. ACM, 2013.

[14] S. Yang, C. Domeniconi, M. Revelle, M. Sweeney, B. Gelman, C. Beckley, and A. Johri. Uncovering trajectories of informal learning in large online communities of creators. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 131–140. ACM, 2015.

Modeling Visitor Behavior in a Game-Based Engineering Museum Exhibit with Hidden Markov Models

Mike Tissenbaum
UW–Madison
225 North Mills St
Madison, WI
miketissenbaum@gmail.com

Vishesh Kumar
UW–Madison
225 North Mills St
Madison, WI
vishesh.kumar@wisc.edu

Matthew Berland
UW–Madison
225 North Mills St
Madison, WI
mberland@wisc.edu

ABSTRACT

Research has shown that supporting tinkering and exploration promotes a wide range of STEM related literacies. However, the open-endedness of tinkering environments makes it difficult to know whether learners' exploration is productive or not. This is especially true in museum spaces, where dwell times are short and facilitators lack a history of engagement with individual visitors. In response, this study uses telemetry data from Oztoc – an open-ended exploratory tabletop exhibit in which visitors embody the roles of engineers who are tasked with attracting and cataloging newly discovered aquatic creatures by building working electronic circuits. This data is used to build Hidden Markov Models (HMMs) to devise an automated scheme of identifying when a visitor is behaving productively or unproductively. Evaluation of our HMM was shown to effectively discern when visitors were productively and unproductively engaging with the exhibit. Using a Markov model, we identify common patterns of visitor movement from unproductive to productive states to shed light on how visitors struggle and the moves they made to overcome these struggles. These findings offer considerable promise for understanding how learners productively and unproductively persevere in open-ended exploratory environments and the potential for developing real time supports to help facilitators know how and when to best engage with visitors.

Keywords

Learning analytics, museums, interactive tabletops modeling.

1. INTRODUCTION

While there is evidence that digitally-augmented museum spaces can enhance science learning [36, 11], there is increased interest in how less-structured, open-ended designs can support new forms of STEM-based (science, technology, engineering, and math) reasoning and collaboration [18, 19]. Tinkering, in particular, often characterized by playful, experimental, iterative styles of engagement, and iterative, investigative processes of learning and discovery, has shown considerable promise in helping novices develop engineering and computer science literacies [5, 26].

Tinkering is an ideal complement to the kinds of learner-centered constructivist pedagogy found in many hands-on science museums [1]; however, in the open-ended and exploratory tasks that typify tinkering, assessment and feedback is particularly difficult [8]. This is especially true in museum environments, as visitors often do not have the expertise or confidence to conduct the coherent, in-depth investigations required to answer their questions on their own [2]. As such, within open-ended environments there is a growing need to develop methods for understanding learners' tinkering and exploration.

Digitally mediated museum spaces, when properly instrumented, can capture data on visitors' tinkering and experimentation in real-time (known as telemetry data), allowing researchers to identify and analyze temporal patterns in visitor interactions. We can then begin to investigate which patterns might be classified as productive (e.g., moving towards the broader learning goals of the exhibit) or unproductive (e.g., [23]). However, by their very nature, productive and unproductive states within open-ended tinkering activities are inherently difficult to classify.

One approach to understanding the state of a learner is through Markov Modeling [4]. Markov modeling is used to characterize patterns of sequential activity, but first-order Markov models only consist of sequences of known states, and we are often more interested in more complex relationships than just sequences of concrete data. One approach to finding hidden states in learners' activities is the use of Hidden Markov Models (HMM – [25]). Applying HMM to learning processes allows us to consider a learner as being in one of a fixed set of ("hidden") states at any moment in time. These models, are particularly well suited for museums as individual visitors' states are particularly hard to capture and pre- and post-tests are problematic if we want to ensure a naturalistic setting [9]. In response, the paper advances a research trajectory in which we attempt to highlight productive and unproductive patterns of visitor interactions by mining their telemetry data from an interactive tabletop exhibit at a large urban interactive science museum. In particular, this research addresses the following questions: *1) Can a Hidden Markov Model accurately predict if visitors are productively or unproductively engaged in an open-ended museum activity? 2) Can we identify the patterns of exploration and tinkering visitors engage in when they move from unproductive to productive states?*

2. BACKGROUND & PRIOR WORK

Within the context of this study, it is important to understand what we consider to be "productive" or "unproductive" patterns of practice. Within the learning sciences, there is interest in practices that can be considered productive for novices who are learning computer sciences and engineering [5]. With its focus on the *processes* of creative and improvisational exploration and making, tinkering is recognized as a means for developing a wide range of STEM literacies [22, 13]. Tinkering is predicated on engaging learners in activities centered on the use of scientific tools, processes, and phenomena to explore a problem space through experimentation, trial and error, and refinement [6, 10, 5].

With tinkering's focus on open exploration and learner-defined goals, understanding how and when a learner is engaged in productive tinkering is a challenge. For instance, making mistakes in "traditional" learning environments is often viewed as failure, but in tinkering environments, failure is not only tolerated but celebrated [26]. At their core, tinkering-focused environments enculturate the notion that learners should be allowed to persevere through initial struggles. However, it is not simply that learners

persist, but *why they are persisting* and *how they are persisting* [27]. With persistence, it is critical that learners actively move towards new solutions or problem conceptualizations, or they risk getting stuck in cycles of unproductive perseverance [23].

In museum settings, understanding when visitors are engaging in productive versus unproductive practices *and* having museum facilitators monitor these states is a challenge. This is especially true in open-ended exploratory exhibits in which multiple visitors can engage and leave at different times (rather than having well-defined beginning and end points) and can interact with the exhibit at multiple granularities (e.g., alone, in groups, or simultaneously with strangers). However, if we can develop ways for capturing visitors' hidden productive and unproductive states, we open up the possibility for understanding underlying patterns in their tinkering and learning and providing critical information to researchers, designers, and museum facilitators.

2.1 Tabletop Interfaces and Engineering

There is significant research into the role the "programming" environment plays in supporting novices in exploring and tinkering when learning computer science and engineering [20, 5]. Tangible engineering platforms, such as "snap together circuits" (e.g., *snapcircuits.net*), allow novices to physically manipulate objects as they tinker and explore engineering concepts, providing clear feedback on their process (with pieces clearly fitting together, or lighting up when properly connected). Such interfaces can reduce learner overhead, freeing them to focus on exploration.

With their ability to support multiple visitors simultaneously and in promoting social interactions, interactive tabletops are increasingly used in science and engineering museum research [9, 1]. In general, interactive tabletops are well suited for supporting engineering practices as they promote greater co-awareness of peers' work [35], and can provide increased opportunities for others to monitor and provide feedback [20, 33]. The addition of tangible blocks (blocks that are recognized by the tabletop when placed on its surface) can further support visitors' engagement with engineering practices by allowing them to quickly try out ideas [16] and more generally explore and tinker.

While tabletops are great for supporting collaborative engineering learning, they can make it more difficult for museum explainers to know the state of tinkering of any one visitor. Similar to the problems teachers face with laptop lids [29], the flat surface of the multitouch tabletop can obscure visitors' interactions, forcing explainers to "hover" in order to know what visitors are doing. Even if explainers do hover, keeping track of multiple visitors' states manually (to know when and where they are needed) would be nearly impossible. In response, we need to develop models that can give us insight into visitor sates, particularly in real-time.

2.2 Markov and Hidden Markov Models

A Markov decision process (MDP) is defined by its state set S, and transition probabilities P [41] – assuming identical actions between states, and identical rewards for each transition. This is represented as a graph, called a Markov Model, which depicts that given a state *s*, the probability of transitioning to any of the other states *s'* is T(s, s'). In a Markov model, transition probabilities are calculated given a sequence of user states. Calculating (and then visualizing) the likelihood of a transition between states has many potential uses: identifying optimal action sequences in Intelligent Tutoring Systems towards success and using these to provide hints to users [3]; or classifying and identifying common student errors and technical problems to reduce their occurrence [15].

Hidden Markov Models (HMMs), as their name suggests, are Markov Models of *hidden* states. These are not directly observed in the input sequences, but, rather, they exist as aggregated "descriptions" of a user's visible states or "action events" [17]. These have been used to classify users through their navigation or content access patterns [12] and characterize student behaviors in computer-based inquiry learning environments [17]. HMMs require: an input sequence of visible states; an initial transition table providing a starting estimate for the transition probabilities between the hidden states; and an emission table with the probabilities of each of the visible states given each hidden state. Initialization and verification for an HMM-based learning model is an important step, as inappropriate initialization might result in the model getting stuck in local minima [7]. After appropriate initialization via the transition and emission tables, the HMM labels each input state with the corresponding hidden states, and gives the transition probabilities between the hidden states.

3. DESIGNING AN OPEN-ENDED TABLETOP ENGINEERING EXHIBIT

3.1 The *Oztoc* Exhibit

In order to address our research goals, we are building upon an existing multitouch tabletop exhibit at a large urban science museum. The exhibit, named *Oztoc* [19], situates visitors as electrical engineers called in to help fictional scientists who have discovered an uncharted aquatic cave teeming with never-before documented species of aquatic life (Figure 1). The creatures who live in this cave are bioluminescent, and visitors are asked to help design and build glowing "fishing lures" to attract the "fish" so that scientists can better study them. Visitors place wooden blocks, which act as electrical components (i.e., batteries, resistors, Light Emitting Diodes or LEDs, and timers), on the interactive table to create simple circuits (which the table recognizes the blocks via fiducial symbols – see Figure 1).

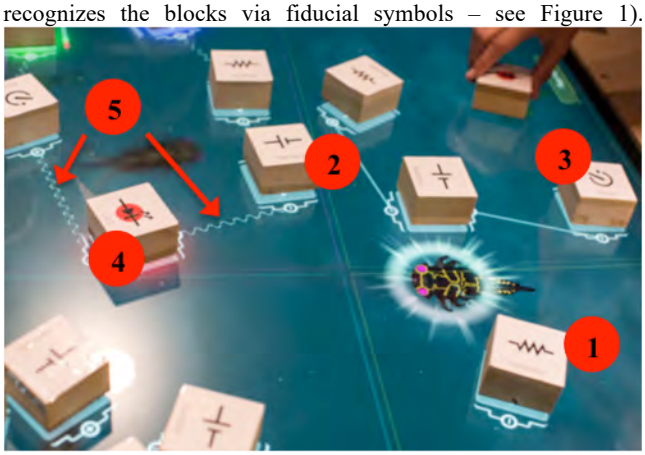


Figure 1. Visitors assemble virtual circuits using wooden blocks that represent resistors (1), batteries (2), timers (3), and different colored LEDs (4). Visitors make circuit connections (depicted as lines on the tabletop - 5) by bringing the positive and negative terminals of the blocks (augmentations displayed by the table) in contact with one another. Creating a successful circuit (one that has the correct ratio of resistors, batteries, and LEDs) causes LEDs to glow and lures creatures attracted to it for cataloging.

Oztoc's narrative aims to give learners a situated context in which to engage in engineering practices. To avoid many of the problems of other engineering and making exhibits [19], we wanted *Oztoc* to give visitors some freedom in choosing their own

goals (e.g., which types of fish to target) while still giving them a common set of materials and processes.

4. METHODOLOGY AND VISITORS

Oztoc is installed in an enclosed exhibit space just off the main floor of a large urban science center. A lollipop sign just outside the exhibit space indicates when videotaping will take place in the exhibit, allowing visitors to decide to enter or to return when data collection is not active. Researchers were present for technical support to museum staff only. Video data was collected via cameras placed in the exhibit space, audio from a boundary microphone, and telemetry data using the ADAGE system [31].

Visitors in this study come from a wide range of backgrounds and SES. Visitors were also multi-generational and came to the exhibit alone, as families, and in large groups.

4.1 Establishing Visitor Start and Stop Times

Unlike many other exhibits, *Oztoc* does not have pre-determined start and stop events (such as the beginning or end of a simulation or game) – it is a continual process in which visitors enter and leave, often at different times. Therefore, in order to accurately separate visitors' sequences of activities, we developed a method for determining when visitors entered or exited the exhibit. Given all actions performed at each of the table's four "zones" over a single day, we found that if a zone was inactive and empty over a set period of time – the "inactivity interval" (InI), the next event in that zone indicated a new visitor. We evaluated an InI ranging from 10-120 seconds, and the InI did not change significantly between 45-120 seconds. As such, we validated the 45-second InI with hand-labeled data. Our 45 second InI achieved full accuracy for the 2-hour sample of video data that we hand-labeled.

4.2 Coding Visitor Events

We needed to establish a granularity of the telemetry data that would allow us to understand the state of visitors' tinkering at any moment. Based on previous research on visitors' interactions with the exhibit [19], we chose to look at the events when visitors successfully created a circuit (denoted in the logs as *MakeCircuitCreate*). This state was particularly useful as a circuit was logged in ADAGE *even if the circuit "didn't work"* (i.e., the LEDs were not supplied correct voltage), giving us insight into visitors' process exploring different circuit configurations, solution states, and goals. By leveraging visitors' histories at the table, we could mine for more complex relationships between their current circuit, previously made circuits, and those made by others at the table since their arrival. We then automatically coded each visitors' MakeCircuitCreate event using four binary codes (see Table 1).

Table 1. Binary codes for MakeCircuitCreate events

Marker	Code	Description
Is the circuit **c**omplex?	S/C	The completed circuit has 3+ components
Does the circuit **w**ork?	N/W	The circuit successfully lights up
Is the circuit **u**nique for self?	R/U	This is the first time the visitor has made this circuit
Is the circuit unique at the table?	E/O	No one else at the table has made a circuit with the same set of components

4.2.1 Is the circuit complex? (coded S or C)

Earlier analysis of visitors' interactions with the exhibit showed that most visitors (if they made *any* circuits) only made the basic three-component circuit (one LED, one resistor, and one battery) [34]. As such, the building of a complex (more than three component) circuit was a key indicator that visitors were trying out more complex configurations. If a circuit had three or less components we scored it an **S** (*indicating it was "simple"*), any circuit that had more than three components was scored a **C** (*indicating it was a complex circuit*). It is important to note that this code is not concerned with *whether or not the circuit works*, only the number of components used.

4.2.2 Does the circuit work? (coded N or W)

Understanding the relationship between the individual components and making a working circuit is a critical factor in determining the success of an exploration. As such, each completed non-working circuit was coded with an **N** and each completed working circuit with a **W**.

4.2.3 Is the circuit unique for self? (coded R or U)

Because problem solving through tinkering is characterized by exploration and iteration [26], we coded if a circuit created by a visitor was "unique" for them (i.e., had they constructed the exact same circuit earlier). A visitor who received a **W** on the *does the circuit work* code might seem to be engaging in productive tinkering; however, if they are simply repeating their first circuit over and over, this might indicate a failure to try out new ideas or expand their problem definition. To mark if a visitor's circuit was unique we coded it with a **U**, if it was a repeat of a past circuit we assigned it an **R.**

4.2.4 Is the circuit unique at the table?

Finally, *Oztoc* is designed to support visitors in collaborating with and building off others' to advance their own exploration. This use of others' constructed artifacts as a basis for one's own work has been termed "echoing" and has been shown to be an important part in open-ended and exploratory tinkering [34]. We considered a circuit to be an echo if it had the same number of each component type (battery, resisters, and LEDs). If a visitor's circuit echoed of one of their peers', we assigned it an **E** (for echo); if the circuit was unique to the table, we assigned it an **O** (for original).

The process described above resulted in every *MakeCircuitCreate* event for each visitor receiving an easily interpretable four-digit code. For instance, a *MakeCircuitCreate* that was assigned a code of **SWRO** means that it was a simple (**S**), working circuit (**W**) that was a repeat of a past circuit made by the visitor (**R**), but had not been created by anyone else at the table since this visitor started playing (**O**). These codes provided a rich and detailed source of data for passing into a Hidden Markov Model to see if we could identify if visitors were productive or unproductive at any point during their engagement with the exhibit. Since the *MakeCircuitCreate* events were chronologically ordered and separated per visitor, we could further examine which created circuits led to important state shifts.

4.3 Coding for productive behaviors

Using the coded descriptions of the circuits created by the visitors, we wanted to make an HMM that identifies when a visitor was behaving "productively", or not. For this purpose, building off of previous research [19], two members of the research team discussed and identified patterns of *MakeCircuitCreate* that were indicative of productive and unproductive tinkering.

One of the key patterns identified focuses on visitors trying out new circuit configurations to fix errors in their existing circuits or to develop new circuits (denoted by a **U** in codes). For instance, if a visitor attempted a few different non-working circuits – seen as a sequence of **SNUO**, **SNRO**, **SNUO, SNUO** (with the second circuit being a duplicate of a past circuit) – the sequence seems to indicate that while the visitor's circuits do not work (indicated by the **N**s), they are trying out new approaches and expanding their exploration. This sequence of activities was coded as productive behavior. If the visitor's continued exploration results in cycle of repeated circuits coded with **R**s (repeats) or did not eventually make a working circuit (coded with a **W**), we coded these actions as falling into unproductivity, as the visitor seems to have failed to figure out how to make a working circuit.

Similarly, a visitor might make a working circuit (indicated by a **W** in their circuit code) and repeat it over and over again (e.g., a series of circuits such as **SWRO**, **SWRO**, **SWRO**). This would seem to indicate that the visitor is repeating past success and is failing to consider new problem spaces or avenues for exploration.

A change of **SNRO** to **SNOE** – trying a new (**U** = self-unique) circuit that someone else on the table has made (**E** = table-echo), might be an attempt at gaining understanding by looking at what other visitors are doing – and was coded as productive depending on how many failed attempts the visitor had already made.

With this understanding, the first two authors first coded 200 circuit creates, and established reliability with 91% agreement. They then coded 644 of the (total of 3952) circuits made in player one's zone (one of the four game quadrants) on the table.

4.4 Training the Hidden Markov Model

We used our manually coded states to calculate appropriate values for the emission table for our HMM. The emission table was calculated by seeing how often a certain circuit code was marked as productive (or unproductive) as a proportion of all the circuits coded with the same hidden state. For instance, of all the circuits coded as productive, 5.6% of those were coded as CWUO and 6.49% were coded as CWUE (from the list of 16 circuit-codes), these values were then used to populate the HMM emission table.

We needed to identify when new visitors started playing at the table to ensure that the new visitors circuits were not considered as a continuation of earlier visitors. To do this we added events (a **0000** code) in the sequence of circuit-codes to signify new visitors. This brought up the question of whether the HMM should code new visitors as unproductive, productive, or another state altogether. To be able to show what state people tended to leave and begin at in the final transition table, we chose to make the visitor change a distinct state in our HMM even though it was not a hidden state, and is equivalent to a direct observation.

We used Python's hmmlearn package to create our HMM, which has the limitation of only looking for local optima in calculating the probabilities of transitioning from one hidden state to another. To account for this, different initial transition table values were tried. Results showed that the HMM stably converged to the final transition table (Figure 3).

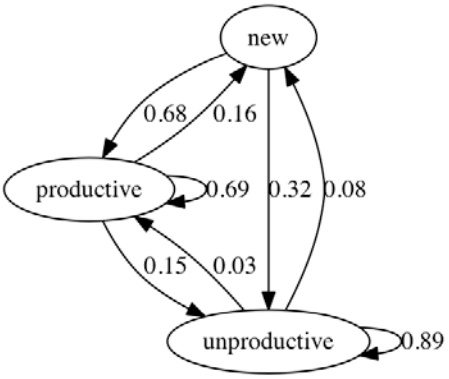


Figure 3. HMM for productive/unproductive states in Oztoc

5. FINDINGS

This study has two important findings, with the first finding acting as the scaffold for the second: First, the recognition of when visitors are engaged in productive or unproductive exploration; and second, the understanding of which sequences of events typically lead visitors from prolonged (at least three) consecutive unproductive states to a productive state.

5.1 Running HMM on Visitors' Circuits

The result of the HMM's final transition table revealed several interesting results (Figure 3). The HMM model shows that the probability of a new visitor beginning productively is 68%, versus 32% for beginning unproductively. Being unproductive appears to be a more stable state than being productive (89% versus 69%, respectively), and moving from unproductivity to productivity is also rarer than the reverse (3% versus 15%). The model also shows that the chances of leaving the table while being productive is higher than of leaving while unproductive (16% versus 8%).

To validate the predictive accuracy of the HMM's classification we used a general agreement score, the calculated the area under the curve (AUC) of the model's receiver operating characteristic (ROC) and Cohen's Kappa as compared to our 644 hand-coded labels. Our HMM had 94% agreement, scored an ROC/AUC score of 0.79, and a Cohen's Kappa of 0.59, which were

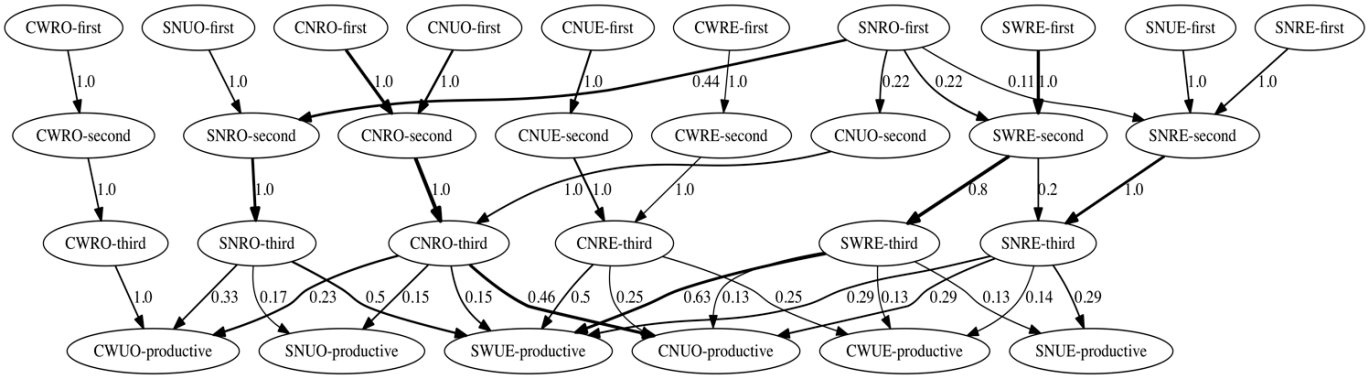


Figure 4. Markov model for visitors who transition from three consecutive unproductive states to a productive state

satisfactory measures to consider the HMM's coding reliable.

5.2 Developing Markov Models of Moving from Unproductive to Productive States

After the HMM tagged the circuits as productive or unproductive, we wanted to understand what patterns of activity preceded visitors becoming productive. We were particularly interested in sequences in which visitors struggled (had several unproductive moves) and then moved to a productive state. For this, we built a list of when a visitor had three consecutive unproductive circuits immediately followed by a productive circuit. We pruned the sequences that only happened once (as they were uninformative).

Once we had a list of the 4 step chains, we made a Markov model depicting the sequences of actions visitors followed when moving from unproductive to productive (Figure 4). This model also showed the likelihood that a visitor making a certain coded circuit would make another specific circuit next. The thickness of the lines between nodes indicates how many times a path occurred.

6. DISCUSSION

This paper outlined how the combination of Hidden Markov Models (HMMs) and Markov chains could be used to effectively predict when visitors were engaging productively or unproductively in an open-ended, exploratory museum exhibit. A closer examination of the HMM revealed several unexpected visitor behaviors. Visitors more often than not (68%) begin productively, but are less likely to stay productive (69%) than unproductive (89%) once in that state (Figure 3). The first finding is not entirely surprising, as our model considers open, thoughtful exploration as productive and it is hard to consider a visitor's "first move" as anything more than a first "exploratory step". This view is partially validated by the lower likelihood of staying productive – indicating many visitors fail to make thoughtful adjustments to their tinkering or explore new definitions of the problem space. This is compounded by instances where visitors make a successful circuit then "settle into" making the same circuit over and over. These findings are supported by the high percentage of visitors who either stay unproductive (89%) or leave the exhibit (8%). It should be noted that 69% is still a very high number of visitors staying productive and is probably further understated by the "first circuit" effect described above.

Another interesting finding is the high likelihood of leaving the table while being productive (16% compared to leaving the table while unproductive – 8%). On the surface this is surprising, as one would expect visitors to give up due to frustration more often than while 'succeeding'. The results may indicate that visitors who "figure out" multiple facets of the exhibit continue to engage productively until they leave – some of these effects have been covered in other research on this project [19]. Another possible explanation is that visitors started to engage in productive behaviors (such as trying something new that they had not done before or echoing the work of another visitor) that didn't immediately result in positive feedback from the system (e.g., capturing a fish) and they gave up.

When looking at the Markov model of unproductive to productive states we uncovered several interesting sequences (see Figure 4). For instance, unproductive circuits coded as **CNUO** (complex, not-working, unique, original) always went to **CNRO** (complex, not-working, repeated, original), followed by another **CNRO**, which finally led 15% of the time to a productive **SWUE** – a simple, working circuit that they had never made earlier, but had been made on the table in front of them by someone else! This is an interesting phenomenon – that a visitor, after some initial failures at making working circuits with a high level of complexity, likely saw a simple working circuit made by someone else, and then switched to echoing that circuit. The ability to see the work of others helped them overcome their own unproductive exploration. We see similar patterns in the Markov chain sequences **SNUO** -> **SNRO** -> **SNRO** -> **SWUE**; and **SNUE** -> **SNRE** -> **SNRE** -> **SWUE**, highlighting the role that making the work of others engaged in parallel tasks visible can serve in helping visitors move from unproductive to productive states.

7. CONCLUSIONS AND NEXT STEPS

Tinkering and exploration are powerful ways for learners to engage in science and engineering practices [24]; however, supporting leaners to productively engage in open-ended learning is inherently difficult, especially in museums [13]. Much of this has to do with the inherent chaos of the museum environment – hundreds (even thousands) of visitors interact with an exhibit in a day, coming and going at different times, and with different expectations and goals. For facilitators in exploratory exhibits, keeping track of the flow of participants and the state of their individual and collective tinkering efforts is nearly impossible.

This paper illustrates how data mining and analytics can help disambiguate the actions of visitors in such exhibits and uncover the hidden states of their tinkering. In addition to shedding light into how visitors productively and unproductively tinker, this work holds considerable potential for developing new ways to support facilitators. Knowing when and how visitors are engaging in unproductive exploration can help us develop complementary applications to help facilitators know when and how they are most needed. Knowing how visitors tend to move from unproductive to productive states can further guide us in developing strategies and scaffolds to help facilitators better engage with visitors.

While tablet applications have been used to provide added contextual information and alert museum facilitators about the visitors' interactions with exhibits in real-time [30], they have done so only using surface features, without understanding visitors' exploration 'states'. By uncovering the particular ways that a visitor is struggling, and understanding the subtle ways they can be "nudged" towards more productive exploration, there is the potential for dramatically influencing visitors' exploration and learning. By interceding at moments where visitors are struggling or are likely to give up, we may increase visitors dwell time, which has been shown to increase their collaboration with others, and domain learning [9]. In response, we are developing a tablet application that uses our models to support facilitators in real-time to understand how such applications compare to approaches that rely only on surface measures and unmodeled log data.

8. REFERENCES

[1] Allen, S. (2002). Designs for learning: Studying science museum exhibits that do more than entertain. *Sci. Ed. 88*, S1 (2004), S17-S33.

[2] Allen, S., & Gutwill, J. P. (2009). Creating a program to deepen family inquiry at interactive science exhibits. *Curator: The Museum Journal*, 52(3), 289-306.

[3] Barnes, T., & Stamper, J. (2008, June). Toward automatic hint generation for logic proof tutoring using historical student data. In *Intelligent tutoring systems* (pp. 373-382). Springer Berlin Heidelberg.

[4] Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*,*37*(6), 1554-1563.

[5] Berland, M., Martin, T., Benton, T., Smith, C. P., & Davis, D. (2013) Using Learning Analytics to Understand the Learning Pathways of Novice Programmers. *Journal of the Learning Sciences 22*(4), 564-599.

[6] Bevan, B., Gutwill, J. P., Petrich, M., & Wilkinson, K. (2015). Learning through STEM-rich tinkering: Findings from a jointly negotiated research project taken up in practice. *Science Education, 99*(1), 98–120.

[7] Bicego, M., & Murino, V. (2004). Investigating hidden Markov models' capabilities in 2D shape classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *26*(2), 281-286.

[8] Blikstein, P. (2013, April). Multimodal learning analytics. In Proceedings of the third international conference on learning analytics and knowledge (pp. 102-106). ACM.

[9] Block, F., Hammerman, J., Horn, M., Spiegel, A., Christiansen, J., Phillips, B., ... & Shen, C. (2015, April). Fluid Grouping: Quantifying Group Engagement around Interactive Tabletop Exhibits in the Wild. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (pp. 867-876). ACM.

[10] Dorn, B., & Guzdial, M. (2010). Learning on the job: Characterizing the programming knowledge and learning strategies of Web designers. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 703–712). New York, NY: ACM.

[11] Feder, M. A., Shouse, A. W., Lewenstein, B., & Bell, P. (Eds.). (2009). *Learning Science in Informal Environments: People, Places, and Pursuits.* National Academies Press.

[12] Fok, A. W., Wong, H. S., & Chen, Y. S. (2005, July). Hidden markov model based characterization of content access patterns in an E-Learning environment. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on* (pp. 201-204). IEEE.

[13] Gutwill, J. P., & Allen, S. (2010). Facilitating family group inquiry at science museum exhibits. *Science Education*, *94*(4), 710-742.

[14] Gutwill, J. P., Hido, N., & Sindorf, L. (2015). Research to practice: Observing learning in tinkering activities. *Curator: The Museum Journal*, *58*(2), 151-168.

[15] Heathcote, E. A., & Prakash, S. (2007). What your learning management system is telling you about supporting your teachers: monitoring system information to improve support for teachers using educational technologies at Queensland University of Technology.

[16] Hornecker, E., & Buur, J. Getting a grip on tangible interaction: a framework on physical space and social interaction. In *Proc. CHI 2006,* ACM Press (2006), 437-446.

[17] Jeong, H., Gupta, A., Roscoe, R., Wagster, J., Biswas, G., & Schwartz, D. (2008, June). Using hidden Markov models to characterize student behaviors in learning-by-teaching environments. In *Intelligent Tutoring Systems* (pp. 614-625). Springer Berlin Heidelberg.

[18] Land, S. M. (2000). Cognitive requirements for learning with open-ended learning environments. *Educational Technology Research and Development*,*48*(3), 61-78.

[19] Lyons, L., Tissenbaum, M., Berland, M., Eydt, R., Wielgus, L., & Mechtley, A. (2015, June). Designing visible engineering: supporting tinkering performances in museums. In *Proceedings of the 14th International Conference on Interaction Design and Children* (pp. 49-58). ACM.

[20] Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch Programming Language and Environment. *ACM Transactions on Computing Education (TOCE)*, vol. 10, no. 4 (November 2010).

[21] Martin, L. (2015). The promise of the Maker Movement for education. *Journal of Pre-College Engineering Education Research*, *5*(1), 30-39.

[22] Martinez, S. L., & Stager, G. (2013). Invent to learn: Making, tinkering, and engineering in the classroom.

[23] McFarlin, D. B., Baumeister, R. F., & Blascovich, J. (1984). On knowing when to quit: Task failure, self-esteem, advice, and nonproductive persistence. *Journal of Personality*, *52*(2), 138-155.

[24] Petrich, M., Wilkinson, K., & Bevan, B. (2013). It looks like fun, but are they learning. *Design, make, play: Growing the next generation of STEM innovators*, 50-70.

[25] Rabiner, L. R., & Juang, B. H. (1986). An introduction to hidden Markov models. *ASSP Magazine, IEEE*, *3*(1), 4-16.

[26] Resnick, M., & Rosenbaum, E. (2013). Designing for tinkerability. *Design, make, play: Growing the next generation of STEM innovators*, 163-181.

[27] Ryoo, J. J., Bulalacao, N., Kekelis, L., McLeod, E., & Henriquez, B. (2015). Tinkering with "Failure": Equity, Learning, and the Iterative Design Process. In *annual FabLearn conference. Palo Alto, CA: Stanford University.*

[28] Schweingruber, H., Keller, T., & Quinn, H. (Ed.). *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*: National Academies Press, 2012.

[29] Sharples, M. (2013). Shared orchestration within and beyond the classroom. *Computers and Education*, *69*, 504-506.

[30] Slattery, B., Lyons, L., Pazmino, P. J., Silva, B. L., & Moher, T. (2014). How interpreters make use of technological supports in an interactive zoo exhibit. *In 11th International Conference of the Learning Sciences (ICLS 2014)*.

[31] Stenerson, M. E., Salmon, A., Berland, M., & Squire, K. Adage: an open API for data collection in educational games. In *Proc, SIGCHI Play 2014*, ACM Press (2014), 437-438.

[32] Sutton, R. & A. Barto. Reinforcement Learning: An Introduction, 1998, The MIT Press, Cambridge, MA.

[33] Tissenbaum, M., Berland, M., & Lyons, L. (*in review*). CCLM Framework: Understanding Collaboration in Constructionist Tabletop Learning. *International Journal of Computer Supported Collaborative Learning*.

[34] Wielgus, Tissenbaum & Berland (*in review*) Echoes paper

[35] Xambó, A., Hornecker, E., Marshall, P., Jorda, S., Dobbyn, C., & Laney, R. (2013). Let's jam the reactable: Peer learning during musical improvisation with a tabletop tangible interface. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 20(6), 36.

[36] Yoon, S. A., Elinich, K., Wang, J., Schoonveld, J. B., & Anderson, E. (2013). Scaffolding informal learning in science museums: How much is too much?. *Science Education*, 97(6), 848

Learning Curves for Problems with Multiple Knowledge Components

Brett van de Sande
Pearson Eduction
brett.vandesande@pearson.com

ABSTRACT
Learning curves have proven to be a useful tool for understanding how a student learns a given skill as they progress through a curriculum. A learning curve for a given Knowledge Component (KC) is a plot of some measure of competence as a function of the number of opportunities the student has had to apply that KC. Consider the case where each problem-solving step is recorded by, for instance, by an intelligent tutoring system. In this case, one normally assigns a unique KC to each problem-solving step and the construction of the associated learning curves is straightforward. On the other hand, many online homework systems only evaluate the student's final answer to a problem. In that case, the student has generally applied a number of KCs to find the answer and their performance on the problem is some composite of their mastery of all of the requisite KCs. In this paper, we propose a simple method for generating learning curves for multiple-KC problems that is independent of any particular theory of learning. In the case where there is only one KC per problem, the method reduces to the ordinary learning curves. We demonstrate this method using a set of artificially generated student data.

Author Keywords
Learning Curves, Knowledge Components

ACM Classification Keywords
I.2.6 Learning: Knowledge acquisition

INTRODUCTION
The increased use of online homework systems and intelligent tutor systems (ITS) means that ever-increasing amounts of student log data is available for analysis. This data can be used to answer two important questions: what skills are students learning and how quickly are they learning them? To be more precise, we can equate skills with Knowledge components (KCs): small bits of information needed to solve a problem [11, 3]. KCs generally have some sort of pre-requisite relations: For example, you cannot apply the area of a circle formula $A = \pi r^2$ unless you first know the definition of "radius of a circle." However, aside from prerequisites, a KC can, by definition, be mastered independently from other KCs. This definition assumes that KCs are *context independent*. That is, the student's ability to apply that KC correctly or quickly does not depend on the particular problem the student is solving or the other KCs needed to solve that problem.

Since KCs are *defined* to have these properties, then it remains to be seen whether, and in what cases, they are a useful description of skill acquisition. One way to determine how well the KC picture is working is to examine the associated learning curves. If the curves are smooth, increasing/decreasing monotonically (depending on the measure of competence), and independent of context, then the KC picture is working.

Learning curves are a plot of some measure of mastery of a skill as a function of the number of opportunities that the student has had to apply that skill. Possible measures of mastery include:

- number of errors made before correctly applying the KC,
- time taken to correctly apply a KC,
- "assistance score," number of errors plus number of requests for help before completing a step, and
- "correctness", whether the student applied the KC correctly without any preceding errors or requests for help.

In the following, we will use "correctness" as our measure of competence for a given skill.

In a typical Intelligent Tutoring System (ITS), the student enters each problem-solving step into the tutor system. It is natural, in that case, to associate one KC with each student input and it is relatively straightforward to construct the associated learning curves. However, many online homework systems only require the student to enter their final answer to a problems into the system. In this case, a single input is the entire problem and it is natural to associate multiple KCs to each student input.

If multiple KCs are associated with a single input, then the construction of learning curves is more difficult. If the student gets the problem wrong, which KC is responsible? This is sometimes called the "assignment of blame problem" [7,

Table 1. List of definitions and quantities

k, l, m: label representing a KC.

t, u, v: label representing opportunity number for some KC.

p: label representing an exercise.

s: the student.

$P_{t,k}$ is a model parameter representing the probability that a student will apply KC k correctly on opportunity t. $P_{t,k} \in [0,1]$.

$\xi_{s,p}$ is the model-given probability that student s will get problem p correct.

$C_{t,k}$ is the number of students in the dataset who correctly applied KC k on opportunity t.

$I(\boldsymbol{t},\boldsymbol{k})$ is the number of students who got a an exercise containing KCs $\boldsymbol{k} = \{k_1, k_2, \ldots\}$ incorrect where $\boldsymbol{t} = (t_1, t_2, \ldots)$ is a vector of corresponding opportunities. This exercise represents opportunity t_a for the student to apply KC k_a.

$\mathcal{T}_{s,p}$ is the set of KC, opportunity pairs such that problem p is opportunity t for student s to apply KC k.

6, 5]. In the following, a simple method is proposed which addresses the assignment of blame problem while making a minimum of theoretical assumptions, allowing one to construct learning curves for exercises with multiple KCs. Our strategy is to introduce a model where every point on each learning curve is identified as a model parameter. These model parameters, and their associated errors, are then determined by a maximum likelihood fit to student log data. In the case of a single KC per problem/step, this reduces to the usual learning curves.

LEARNING CURVE MODEL

A number of studies have addressed the multiple-KC problem in the context of some model of learning, such as Bayesian Knowledge Tracing or Performance Factor Analysis [2, 4]. In the present work, our goal is simply to construct learning curves using a minimum number of model assumptions. Note that conventional learning curves themselves make two major assumptions:

1. They average over students. This corresponds to a model that does not have any student-specific parameters.

2. They ignore the problem context. This corresponds to a model that does not have any problem-specific parameters.

In fact, the construction of a learning curve is equivalent to fitting the student log data to a model containing a parameter representing each KC and step. In other words, if I define $P_{t,k}$ as the probability that a student will correctly apply KC k at opportunity t, and determine $P_{t,k}$ by fitting to the student log data, then plotting of $P_{t,k}$ versus t is a learning curve for KC k.

This gives us a way forward in the multiple-KC case. We define a model having parameters $\{P_{t,k}\}$. The associated log-likelihood is

$$\log(\mathcal{L}) = \sum_{s,p \in \mathcal{C}_s} \log(\xi_{s,p}) + \sum_{s,p \in \mathcal{I}_s} \log(1 - \xi_{s,p}) \qquad (1)$$

where s is the student, p is the problem, $\mathcal{C}_s$ is the set of problems s got correct, and $\mathcal{I}_s$ is the set of problems s got incorrect. Also, $\xi_{s,p}$ is the model-given probability that student s will get problem p correct.

We will assume that the student must apply *all* of the associated KCs to solve a given exercise correctly. This is sometimes called a "conjunctive model" and is a good approach for typical K-12 math exercises [8]. This means that the total probability of success is the product of the KC probabilities:

$$\xi_{s,p} = \prod_{t,k \in \mathcal{T}_{s,p}} P_{t,k} \qquad (2)$$

where $\mathcal{T}_{s,p}$ is the set of KCs and opportunities such that problem p is opportunity t for student s to apply KC k.

To construct $\mathcal{T}_{s,p}$, one needs a list of KCs associated with each exercise p, sometimes referred to as the "Q-matrix" [10]. In this discussion, we will assume that the Q-matrix is known, perhaps determined by the problem author or a domain expert.

Numerical Calculation

The likelihood given by Eqn. (1) is rather inconvenient for large numerical calculations. Instead, we will introduce variables that aggregate over student and exercise. Define $C_{t,k}$ to be the number of students in the dataset who correctly applied KC k on opportunity t. Likewise, define $I(\boldsymbol{t},\boldsymbol{k})$ to be the number of students who got a an exercise containing KCs $\boldsymbol{k} = \{k_1, k_2, \ldots\}$ incorrect where $\boldsymbol{t}$ is a vector of associated opportunities. This exercise represents opportunity t_a for the student to apply KC k_a. Then, the log-likelihood can be written as

$$\log(\mathcal{L}) = \sum_{t,k} C_{t,k} \log(P_{t,k}) + \sum_{\boldsymbol{t},\boldsymbol{k}} I(\boldsymbol{t},\boldsymbol{k}) \log(1 - \Gamma(\boldsymbol{t},\boldsymbol{k})) \qquad (3)$$

where $\Gamma(\boldsymbol{t},\boldsymbol{k})$ is the probability that a student with opportunity vector $\boldsymbol{t}$ will have success on a problem containing KCs $\boldsymbol{k} = \{k_1, k_2, \ldots\}$. Following Eqn. (2), $\Gamma(\boldsymbol{t},\boldsymbol{k})$ is a product over the associated probabilities:

$$\Gamma(\boldsymbol{t},\boldsymbol{k}) = \prod_a P_{t_a,k_a} \; . \qquad (4)$$

Note that the first term of Eqn. (3) has a much simpler form than the second term. This is due to our use of a conjunctive model. If a student gets an exercise "correct" then we know without ambiguity that they applied all of the associated KCs correctly. However, if they get a problem wrong, then it is not clear which KC is to blame and the associated probabilities must be considered jointly.

Let $\{\widehat{P}_{t,k}\}$ be the model parameters at the maximum likelihood point. $\{\widehat{P}_{t,k}\}$ can be found numerically by maximizing the log-likelihood, Eqn. (3) subject to the constraints

Table 2. KC content of the artificial homework set. Students completed the first eight problems in the given order and the remaining problems in random order; they completed between 15 and 20 problems total.

1	2	3	4	5	6	7	8	9	10
A	A	A	A	B	B	B	B	A	B
11	12	13	14	15	16	17	18	19	20
A	B	AB	AB	AB	AB	AB	AB	AB	AB

$0 \leq P_{t,k} \leq 1$. For convenience, the *Mathematica* function **FindMaximum**, was used to calculate the maximum of $\log(\mathcal{L})$. However, any optimization algorithm that enforces constraints and uses information about the gradient of the function should work as well.

Error analysis

It is important to calculate the standard errors associated with the model parameters. Unlike the single KC per problem case, the model parameters may be strongly correlated and the errors can have unexpected values. In addition, the error analysis can elucidate any cases where the model parameter cannot be determined from the data (we will discuss this further in the conclusion).

Before finding the errors, we need to examine the the maxiumum likelihood point and identify any parameters that lie on the boundaries $\widehat{P}_{t,k} = 0$ or 1. The likelihood function $\mathcal{L}$ is not stationary in these parameters at the maximum likelihood point, so the error analysis cannot be applied to them; they should be not be included in the Hessian matrix below, Eqn (5). In practice, this should not a significant issue, since $\widehat{P}_{t,k} = 0$ or 1 typically occurs when there are just a few student problem-solving instances for a given t and k.

For a maximum likelihood fit, the standard errors associated with the model parameters can determined using the following procedure [1, 9]. First, we find the Hessian matrix associated with $P_{t,k} = \widehat{P}_{t,k}$. The matrix elements of the Hessian are given by

$$\left.\frac{\partial^2 \log(\mathcal{L})}{\partial P_{t,k} \partial P_{u,l}}\right|_{P_{v,m}=\widehat{P}_{v,m}} = -\frac{1}{\widehat{P}_{t,k}\widehat{P}_{u,l}} \sum_{\boldsymbol{t},\boldsymbol{k}} \left.\frac{I(\boldsymbol{t},\boldsymbol{k})\,\Gamma(\boldsymbol{t},\boldsymbol{k})}{(1-\Gamma(\boldsymbol{t},\boldsymbol{k}))^2}\right|_{P_{v,m}=\widehat{P}_{v,m}} . \quad (5)$$

To find the standard error associated with each of the model parameters $\widehat{P}_{t,k}$, we invert the negative of the Hessian matrix and take the square root of the diagonal elements. If this process fails (the Hessian matrix is singular), it is a signal that some of the model parameters cannot be uniquely determined from the given log data. Similarly, if the Hessian matrix is nearly singular, then the associated standard errors will be very large. This will single out any model parameters that cannot be determined from the data.

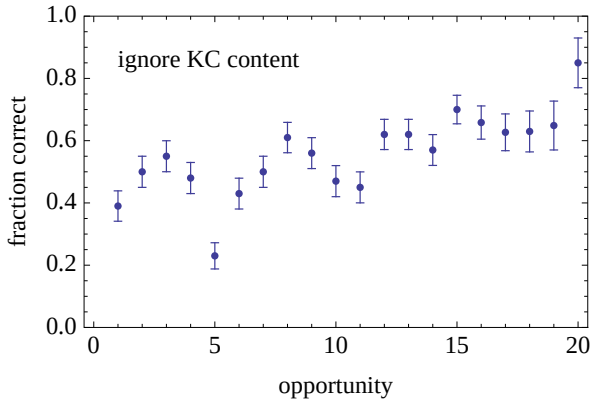


Figure 1. Learning curve for the artificial homework set where we assume each problem has the same single KC. Note the jump after opportunity 4 due to the fact that the first four and second four problems have different KCs.

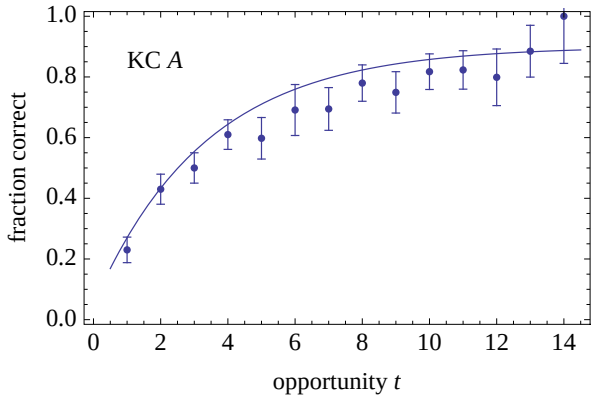


Figure 2. Learning curve for KC A. The solid line is the model used to generate the student data and the points with error bars represent the learning curve determined from the student data using our procedure. Note that the error bars for the last few opportunities are larger, due to student attrition.

APPLICATION TO STUDENT DATA

To illustrate how this model works, we will generate an artificial student performance dataset. Consider a homework assignment of 20 problems that exercise two KCs, A and B as detailed in Table 2. We assume that students progress through the first 8 problems in the given order, but solve the remaining 12 problems in random order, completing between 15 and 20 problems. We assume that student mastery for the KCs is given by the functions $P_{t,A} = 0.9 - 0.85\mathrm{e}^{-0.3t}$ and $P_{t,B} = 0.85 - 0.45\mathrm{e}^{-0.1t}$; see Figures 2 and 3. We use this model to generate a set of outcomes, $\mathcal{C}_s$, $\mathcal{I}_s$, and $\mathcal{T}_{s,p}$, for 100 students.

If we ignore the KC content of the problems, we can plot a naïve learning curve for this student data; See Fig. 1. We see a discontinuity at $t = 4$ due to the change in actual KC content of the problems. The last problems are more difficult, since they involve two skills and so the student performance on them is suppressed.

Next, we use our procedure to generate learning curves and associated errors for this dataset. The results are plotted in

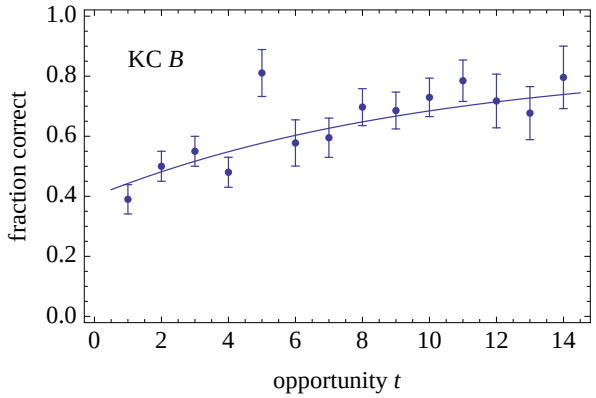


Figure 3. Learning curve for KC B. The high value at $t = 5$ is a statistical fluctuation: as we iincrease the number of students, the model parameters will converge to the solid line.

Figs. 2 and 3. As expected, they agree well with the model used to generate the student data. This shows that our method is working. Note that the error bars can vary considerably from point to point.

CONCLUSION

The primary goal of the approach developed here is to plot learning curves for cases where there are problems (or problem steps) involving multiple KCs. In practice, we find our method to be numerically robust (no problems with local maxima).

However, there is one case where it may fail: if there is a KC that always appears along with another KC for several problems and all the students in the dataset solve nearly the same ordered sequence of problems, then there is no way distinguish between the two KCs for one or more value of t. This will result in a Hessian matrix that is not positive-definite and the matrix inversion will fail. We believe that this situation will rarely arise in practice, since most datasets involve students in multiple courses, and students are generally not forced to solve problems in a specific order.

In this work, we focused on a "conjunctive model" for combining KCs, as this is likely the most appropriate model for typical math and science exercises. Although the basic strategy we present here could be applied to other models (disjunctive, compensatory) for combining KCs, the details of the associated numerical calculation would look rather different.

Obviously, the next step is to apply this approach to real student data. This would require a set of exercises that have been tagged with multiple KCs, where the mix of KCs vary significantly from exercise to exercise. In addition, the student activity would have to fairly heterogeneous, with different students taking different paths through the exercises.

ACKNOWLEDGMENTS

This work was supported by Pearson education. I would like to thank Ilya Golden for reviewing the manuscript.

REFERENCES

1. Edwards, A. W. F. *Likelihood*. Johns Hopkins University Press, 1992.
2. Gong, Y., Beck, J., and Heffernan, N. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In *Intelligent Tutoring Systems*, V. Aleven, J. Kay, and J. Mostow, Eds., vol. 6094 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010, 35–44.
3. Koedinger, K. R., Corbett, A. T., and Perfetti, C. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Sci. 36*, 5 (2012), 757–798.
4. Koedinger, K. R., Pavlik, P. I., Stamper, J., Nixon, T., and Ritter, S. Avoiding Problem Selection Thrashing with Conjunctive Knowledge Tracing. In *Proceedings of the 3 rd International Conference on Educational Data Mining* (2010), 91–100.
5. Nwaigwe, A., and Koedinger, K. R. The Simple Location Heuristic is Better at Predicting Students' Changes in Error Rate Over Time Compared to the Simple Temporal Heuristic. In *Proceedings of the 4th International Conference on Educational Data Mining* (Eindhoven, the Netherlands, 2011), 71–80.
6. Nwaigwe, A., Koedinger, K. R., Vanlehn, K., Hausmann, R., and Weinstein, A. Exploring alternative methods for error attribution in learning curves analysis in intelligent tutoring systems. *Frontiers in Artificial Intelligence and Applications 158* (2007), 246.
7. Ohlsson, S. Towards Intelligent Tutoring Systems that Teach Knowledge Rather than Skills: Five Research Questions. In *New Directions in Educational Technology*, E. Scanlon and T. O'Shea, Eds., no. 96 in Nato ASI Subseries F. Springer Berlin Heidelberg, Berlin, Heidelberg, 1992.
8. Pardos, Z. A., Beck, J. E., Ruiz, C., and Heffernan, N. T. The Composition Effect: Conjunctive or Compensatory? An Analysis of Multi-Skill Math Questions in ITS. In *Educational Data Mining 2008: 1st International Conference on Educational Data Mining, Proceedings*, UNC-Charlotte, Computer Science Dept. (Montreal, Canada, June 2008), 147–156.
9. Pawitan, Y. *In All Likelihood: Statistical Modelling and Inference Using Likelihood*. Oxford University Press, June 2001.
10. Tatsuoka, K. K. Rule Space: An Approach for Dealing with Misconceptions Based on Item Response Theory. *Journal of Educational Measurement 20*, 4 (Dec. 1983), 345–354.
11. VanLehn, K. The Behavior of Tutoring Systems. *Int. J. Artif. Intell. Ed. 16*, 3 (Jan. 2006), 227–265.

A Nonlinear State Space Model for Identifying At-Risk Students in Open Online Courses

Feng Wang and Li Chen
Department of Computer Science
Hong Kong Baptist University, Hong Kong
{fwang, lichen}@comp.hkbu.edu.hk

ABSTRACT

How to identify at-risk students in open online courses has received increasing attention, since the dropout rate is unexpectedly high. Most prior studies have focused on using machine learning techniques to predict student dropout based on features extracted from students' learning activity logs. However, little work has viewed the dropout prediction problem as a sequence classification problem in the consideration that the dropout probability of a student at the current time step can be likely dependent on her/his engagement at the previous time step. Therefore, in this paper, we propose a nonlinear state space model to solve this problem. We show how students' latent states at different time steps can be learned via this model, and demonstrate its outperforming prediction accuracy relative to related methods through experiment.

Keywords

At-risk students; Dropout prediction; Open online courses, Nonlinear state space model

1. INTRODUCTION

With the advent of open online courses, such as MOOC websites Edx, Coursera, Khan Academy, high quality education can easily be accessed by students at low cost. However, although many thousands of participants have enrolled on the online courses, their dropout rate is extremely higher than expected. As reported in [8], the average dropout rate of current MOOCs is approximately 75%.

Identifying at-risk students by predicting their dropout probability thus becomes timely important, given that early prediction can help instructors provide proper support to those students to retain their learning interests. To address this issue, some researchers focused on extract features from students' learning activities (such as watching videos, working on assignments, and posting in or viewing discussion forums) for building machine learning models (like support vector machine (SVM) [9] and logistic regression (LG) [14]). However, they rarely considered that students' learning activities across different time steps (e.g., weeks) might be interrelated and take different weights in making the prediction. For instance, recent activities could be more important to reflect students' engagement degree. If a student actively engages with a course in the current week, it is more likely that s/he will continue to engage with this course in the coming week. Otherwise, if s/he becomes inactive, it may infer that her/his interest in the course is decreased. Recently, though some approaches, such as the one based on Hidden Markov Model (HMM) [2] and that based on Recurrent Neural Network (RNN) [12], have been proposed to model students' states over time, they still suffer from some issues: 1) the estimation of next state depends only on the current state; 2) the estimated states are deterministic that would lead to error propagation in the estimation procedure; 3) the parameters of their models are time-invariant.

In our work, we focus on predicting whether a student will have activities in the coming week. We particularly formulate this issue as *sequential classification* problem, and develop *Nonlinear State Space Model* (NSSM) [1] to solve it. Essentially, NSSM has several advantages. Firstly, it can be used to discover a student's latent state (i.e., *engagement pattern*) to characterize the student's intention to perform certain activities. The student's dropout probability is then computed based on the state estimated for that time. Secondly, relative to HMM and RNN, NSSM takes into account all of the current and previous states to estimate next state. It can also accommodate uncertainty given that the state in NSSM is a set of random variables with *multivariate Gaussian distribution*. Thirdly, the parameters in NSSM are time varying (i.e., being different at different time steps), which makes it more flexible to model students' dynamics.

In short, this paper has two main contributions: 1) we implement Nonlinear State Space Model (NSSM) to address the dropout prediction problem, which particularly models students' latent states varying over time; 2) we conduct experiment to compare our method with related ones including logistic regression (LG), simultaneously smoothed logistic regression (LR-SIM), and RNN with long short-term memory cell (LSTM). It shows that our method is more accurate in identifying at-risk students who tend to drop out.

In the remainder, we first describe related work in Section 2, and then present our methodology in Section 3. In Section 4,

we give experimental results. In Section 5, we conclude our work and indicate its future directions.

2. RELATED WORK

High dropout rate that popularly exists in current MOOCs has driven some researchers to investigate the issue of identifying at-risk students who are likely to quit. They have considered different features to build the prediction model, such as those extracted from clickstream data (e.g., watching a lecture video, posting to discuss forums, submitting an assignment) [2, 5, 6, 9, 14], quiz performance [5, 6, 14], centrality of students in discussion forums [15], and sentiments of discussion forum posts [4].

As for prediction model, some studies have applied support vector machines (SVM) [9], logistic regression (LG) [14], survival analysis techniques like Cox proportional hazard model [15], and probabilistic soft logic (PSL) [13]. However, their common limitation is that they assume a student's dropout probabilities at different time steps are independent, which limits the approach's applicability in practice as usually a student's state at one time can be influenced by her/his previous state.

Alternatively, [6] extended logistic regression model to smooth the dropout probabilities across weeks with the aim to minimize the difference of successing predicted probabilities between weeks. [2] used Hidden Markov Model (HMM) to model student's actions over time, which encodes their behaviour features into a set of mutually exclusive discrete states. [12] adopted Recurrent Neural Network (RNN) model with long short-term memory (LSTM) cells, which is able to encode features into continuous states. However, though RNN may be advantageous against HMM, it inherently suffers from error propagation phenomenon because the estimation of current state depends only on the estimated previous state.

In comparison, in our model, the uncertainty of estimated states is considered by representing the state as random variables drawing from a multivariate Gaussian distribution. What's more, we adopt extended Kalman filter and smoother for state estimation so as to take into account all observed activities in sequence, which makes it different from, and potentially more effective than, HMM and RNN where only states at two consecutive time steps are related.

3. OUR METHODOLOGY

3.1 Problem Statement

As mentioned above, our goal is to estimate the probability that a student stops engaging with a course in the coming week, given her/his learning activities up to the current time step.

The temporal prediction of dropout probability requires us to assemble some features [1] for expressing time-varying behavior of students. Therefore, we extract 28 typical features for each week t, denoted as N dimensional vector $\mathbf{x}_{i,t} \in \mathbb{R}^N$, by considering the seven types of activity [2]. The summarization of these temporal features is listed in Table 1.

[1]Prior to model training, these features are normalized to have mean 0 and variance 1, and the normalization parameters (mean, standard deviation) are used for normalizing the testing set.

Table 1: List of features derived from each student's learning activities by the week t

Features	Description
x_1	The average number of activities per week by the week t.
x_2	The total number of activities in week t.
x_3	The average number of sessions per week by the week t. [3]
x_4	The total number of sessions in week t.
x_5	The average number of active days per week by the week t. [4]
x_6	The total number of active days in week t.
x_7	The average time consumption per week by the week t.
x_8	The total time consumption in week t.
x_9 - x_{15}	The average number of 7 different types of activity per week by the week t.
x_{16} - x_{22}	The total number of 7 different types of activity in week t.
$x_{23} - x_{25}$	The average number of videos watched, wiki viewed and problem attempted per session by the week t respectively.
$x_{26} - x_{28}$	The average number of videos watched, wiki viewed and problem attempted per session in week t respectively.

In consequence, we obtain a sequence $(\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,n_i})$ for each student i across n_i weeks, as well as the corresponding sequence of dropout labels $(y_{i,1}, y_{i,2} \ldots, y_{i,n_i})$. Here n_i represents the number of weeks during which student i has engaged with the course. Formally, for current week t, if there are activities associated to student i in the coming week, her/his dropout label in the week t is assigned $y_{i,t} = 0$, otherwise $y_{i,t} = 1$. We can then treat the dropout prediction task as a *sequential classification* problem, for which the student's latent states evolving over time are not observable directly. As illustrated in Figure 1, as the course progresses, given the student i's features $\mathbf{x}_{i,t}$ for the current week t, and his/her previous state $\mathbf{s}_{i,t-1}$, we want to estimate the student's current state $\mathbf{s}_{i,t}$ and whether s/he will continue engaging with the course in the coming week $y_{i,t}$.

3.2 Nonlinear State Space Model (NSSM)

Specifically, we employ a nonlinear state space model (NSSM) with continuous value states to summarize all the information about a student's past behavior. Formally, let the vector $\mathbf{s}_{i,t} \in \mathbb{R}^K$ ($K \ll N$) be the latent state of student i in the t-th week, which depends on the observed explanatory features $\mathbf{x}_{i,t}$ and her/his previous state $\mathbf{s}_{i,t-1}$, as follows:

$$\mathbf{s}_{i,t} = \mathbf{F}\mathbf{s}_{i,t-1} + \mathbf{G}\mathbf{x}_{i,t} + \mathbf{w}_{i,t} \quad (1)$$

in which the matrix $\mathbf{F} \in \mathbb{R}^{K \times K}$ transforms the previous state into the current state, the matrix $\mathbf{G} \in \mathbb{R}^{K \times N}$ transforms the observed features to reflect the current state, and

[2]The seven types of activity consist of watching lecture videos, working on course's problems, accessing course's modules, accessing course's wiki, posting or viewing course's forum, navigating through courses, and closing course page.
[3]The minimal elapsed time between two separate sessions is set as 60 minutes.
[4]The day that has at leas one activity is treated as an active day.

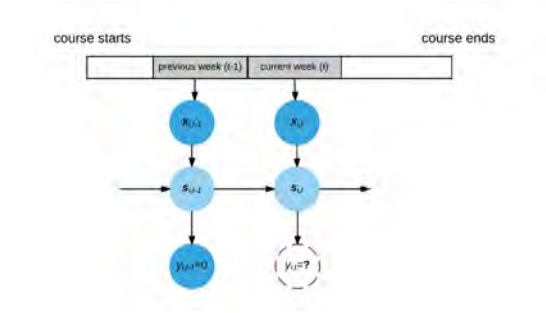


Figure 1: The illustration of MOOCs dropout prediction problem and the graphical state space model. The dark blue signifies an observed variable and the light blue signifies a latent variable.

$\mathbf{w}_{i,t}$ represents a diffusion variable which follows a multivariate Gaussian with mean $\mathbf{0}$ and covariance $\mathbf{Q}_{i,t}$ (i.e., $\mathbf{w}_{i,t} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{i,t})$). Note that the dimension of the state vector K is usually smaller than the dimension of feature vector N. This hyperparameter K controls the complexity of the model, and requires manual tuning to determine its optimal value.

In our work, we aim to infer the dropout probability $\pi_{i,t}$ for student i in week t, which can be represented as logistic regression

$$\pi_{i,t} = \sigma(\mathbf{h}_t^T \mathbf{s}_{i,t} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t}) \quad (2)$$

$$= \frac{1}{1 + \exp(-\mathbf{h}_t^T \mathbf{s}_{i,t} - \boldsymbol{\beta}_t^T \mathbf{x}_{i,t})} \quad (3)$$

where $\mathbf{h}_t \in \mathbb{R}^{K \times 1}$ and $\boldsymbol{\beta}_t \in \mathbb{R}^{N \times 1}$ are two vectors of coefficients for current state variable $\mathbf{s}_{i,t}$ and input feature $\mathbf{x}_{i,t}$ respectively. In this model, the non-stationary of student dynamic is captured by time-evolving state variable $\mathbf{s}_{i,t}$, and time-varying parameters $\mathbf{h}_t$ and $\boldsymbol{\beta}_t$.

3.3 Expectation Maximization

With the nonlinear state space model described in Eqn. 1 and Eqn. 2, we design an Expectation-Maximization (EM) algorithm (see Algorithm 1) that iterates between state estimation (E-step) and parameter estimation (M-step) [11]. The E-step makes use of extended Kalman *filter* and *smoother* to estimate states, and the M-step re-estimates the parameters by maximizing the likelihood of all observed data, in which the state variables of student are replaced by their posteriori values from the extended Kalman smoother.

3.3.1 Expectation Step

In the expectation step, the expected mean of student state $\mathbf{s}_{i,t}$ and its covariance $\mathbf{P}_{i,t}$ are obtained using the extended Kalman filter and smoother. Specifically, given student i's entire $t-1$ weeks' observation sequence $D_i^{(t-1)} = \{(\mathbf{x}_{i,1}, y_{i,1}), (\mathbf{x}_{i,2}, y_{i,2}), \ldots, (\mathbf{x}_{i,t-1}, y_{i,t-1})\}$, the posterior mean and covariance of student state $\mathbf{s}_{i,t-1}$ are supposed be represented by $E(\mathbf{s}_{i,t-1}|D_i^{(t-1)}) = \mathbf{s}_{i,t-1}^{(t-1)}$ and $Cov(\mathbf{s}_{i,t-1}|D_{i,t-1}) = \mathbf{P}_{i,t-1}^{(t-1)}$ respectively. The predicted student state $\mathbf{s}_{i,t}$ and its covariance $\mathbf{P}_{i,t}^{(t-1)}$ for $t = 1, 2, \ldots, n_i - 1, n_i$ can then be defined as:

Algorithm 1 EM algorithm for estimating latent student state and model parameters.

1: Initialize each student's starting state $\mathbf{s}_{i,0}$ and model parameters $\Phi = \{\mathbf{F}, \mathbf{G}, \mathbf{h}_t, \boldsymbol{\beta}_t\}$
2: **repeat**
3: **procedure E-step:**
4: **Extended Kalman filter:** For $t = 1, 2, \ldots, n_i - 1, n_i$, correct the student state $\mathbf{s}_{i,t}$ and its covariance $\mathbf{P}_{i,t}$ by using Eqn. 10 and Eqn. 11 respectively.
5: **Extended Kalman smoother:** For $t = n_i, n_i - 1, \ldots, 2, 1$, smooth the predicted student state $\mathbf{s}_{i,t}^{(t)}$ and covariance $\mathbf{P}_{i,t}^{(t)}$ by using Eqn. 13 and Eqn. 14 respectively.
6: **end procedure**
7: **procedure M-step:**
8: Update parameters of the model Φ via equations from Eqn. 17 to Eqn. 20.
9: **end procedure**
10: **until** converged

$$\mathbf{s}_{i,t}^{(t-1)} = \mathbf{F}\mathbf{s}_{i,t-1}^{(t-1)} + \mathbf{G}\mathbf{x}_{i,t} \quad (4)$$

$$\mathbf{P}_{i,t}^{(t-1)} = \mathbf{F}\mathbf{P}_{i,t-1}^{(t-1)}\mathbf{F}^T + \mathbf{Q}_{i,t} \quad (5)$$

By following the extended Kalman filtering, the nonlinear function $\sigma(\cdot)$ can be approximated by its Taylor series expansion as follows:

$$\pi_{i,t} = \sigma(\mathbf{h}_t^T \mathbf{s}_{i,t} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t}) \approx \sigma(\mathbf{h}_t \mathbf{s}_{i,t}^{(t-1)} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t}) + \mathbf{A}_{i,t}^T (\mathbf{s}_{i,t} - \mathbf{s}_{i,t-1}^{(t-1)}) \quad (6)$$

where

$$\mathbf{A}_{i,t} \triangleq \frac{\partial \sigma(\mathbf{h}_{i,t}^T \mathbf{s}_{i,t} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t})}{\partial \mathbf{s}_{i,t}} = \sigma\left(\mathbf{h}_{i,t}^T \mathbf{s}_{i,t}^{(t-1)} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t}\right) \left(1 - \sigma(\mathbf{h}_{i,t}^T \mathbf{s}_{i,t}^{(t-1)} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t})\right) \mathbf{h}_{i,t} \quad (7)$$

The one-step ahead prediction $\pi_{i,t}^{(t-1)}$ for the dropout probability is computed as:

$$\pi_{i,t}^{(t-1)} = \sigma(\mathbf{h}_t^T \mathbf{s}_{i,t}^{(t-1)} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t}) \quad (8)$$

For the sake of simplicity, we set the state noise covariance as $\mathbf{Q}_{i,t} = q_{i,t}\mathbf{I}$, where the state noise variance $q_{i,t}$ is computed via:

$$q_{i,t} = max\{\mu_{i,t}^{(t)} - \mu_{i,t}^{(t-1)}, 0\} \quad (9)$$

in which $\mu_{i,t}^{(\cdot)} = \pi_{i,t}^{(\cdot)}(1 - \pi_{i,t}^{(\cdot)})$. After receiving a new observation $(\mathbf{x}_{i,t}, y_{i,t})$, the predicted state $\mathbf{s}_{i,t}^{(t-1)}$ in Eqn. 4 and covariance $\mathbf{P}_{i,t}^{(t-1)}$ in Eqn. 5 will be updated as:

$$\mathbf{s}_{i,t}^{(t)} = \mathbf{s}_{i,t}^{(t-1)} + \mathbf{K}_{i,t}\left(y_{i,t} - \sigma(\mathbf{h}_t^T \mathbf{s}_{i,t}^{(t-1)} + \boldsymbol{\beta}_t^T \mathbf{x}_{i,t})\right) \quad (10)$$

$$\mathbf{P}_{i,t}^{(t)} = (\mathbf{I} - \mathbf{K}_{i,t}\mathbf{A}_{i,t})\mathbf{P}_{i,t}^{(t-1)} \quad (11)$$

in which $\mathbf{K}_{i,t}$ is the Kalman gain computed according to [3]:

$$\mathbf{K}_{i,t} = \mathbf{P}_{i,t}^{(t-1)}\mathbf{A}_{i,t}^T\left(\mathbf{A}_{i,t}\mathbf{P}_{i,t}^{(t-1)}\mathbf{A}_{i,t}^T + \mathbf{Q}_{i,t}\right)^{-1} \quad (12)$$

It is worth noting that the predicted state $\mathbf{s}_{i,t}^{(t)}$ and covariance $\mathbf{P}_{i,t}^{(t)}$ in Kalman filter are estimated based on the observation $D_i^{(t)}$ up to week t. We take advantage of extended

Kalman smoother to smooth the estimated states by considering the entire sequence of the student's observations $D_i^{(n_i)}$. The smoothed states could hence be more accurate than the filtered ones. Specifically, the student state $\mathbf{s}_{i,t-1}^{(n_i)}$ and covariance $\mathbf{P}_{i,t-1}^{(n_i)}$ for $t = n_i, n_i - 1, \ldots, 1$ are recursively smoothed as:

$$\mathbf{s}_{i,t-1}^{(n_i)} = \mathbf{s}_{i,t-1}^{(t-1)} + \mathbf{J}_{i,t-1}\left(\mathbf{s}_{i,t}^{(n_i)} - \mathbf{F}\mathbf{s}_{i,t-1}^{(t-1)} - \mathbf{G}\mathbf{x}_{i,t-1}\right) \quad (13)$$

$$\mathbf{P}_{i,t-1}^{(n_i)} = \mathbf{P}_{i,t-1}^{(t-1)} + \mathbf{J}_{i,t-1}\left(\mathbf{P}_{i,t}^{(n_i)} - \mathbf{P}_{i,t}^{(t-1)}\right)\mathbf{J}_{i,t-1}^T \quad (14)$$

where $\mathbf{J}_{i,t-1}$ is the smoothing gain defined as:

$$\mathbf{J}_{i,t-1} = \mathbf{P}_{i,t-1}^{(t-1)}\mathbf{F}^T\left(\mathbf{P}_{i,t}^{(t-1)}\right)^{-1} \quad (15)$$

Note that the initial values $\mathbf{s}_{i,n_i}^{(n_i)}$ and $\mathbf{P}_{i,n_i}^{(n_i)}$ for the smoother are the final estimates of the filter.

3.3.2 *Maximization Step*

At the maximization step, given the observed data D of N students, the likelihood is defined as

$$\begin{aligned}\mathcal{L}(D|\Phi) = &\sum_{i=1}^{N}\sum_{t=1}^{n_i} y_{i,t}\log(\sigma(\mathbf{h}_{i,t}^T\mathbf{s}_{i,t}^{(n_i)} + \boldsymbol{\beta}_t^T\mathbf{x}_{i,t})) \quad (16)\\ &+ (1-y_{i,t})\log(1-\sigma(\mathbf{h}_{i,t}^T\mathbf{s}_{i,t}^{(n_i)} + \boldsymbol{\beta}_t^T\mathbf{x}_{i,t}))\\ &- \frac{1}{2}\sum_{i=1}^{N}\sum_{t=1}^{n_i}(\mathbf{s}_{i,t}^{(n_i)} - \mathbf{F}\mathbf{s}_{i,t-1}^{(n_i)} - \mathbf{G}\mathbf{x}_{i,t})^T\mathbf{Q}_{i,t}^{-1}(\mathbf{s}_{i,t}^{(n_i)}\\ &- \mathbf{F}\mathbf{s}_{i,t-1}^{(n_i)} - \mathbf{G}\mathbf{x}_{i,t}) - \frac{1}{2}\sum_{i=1}^{N}\sum_{t=1}^{n_i}\log|\mathbf{Q}_{i,t}|\end{aligned}$$

By using the posterior hidden state variables $\mathbf{s}_{i,t}^{(n_i)}$ from Kalman smoother, the optimal parameters $\Phi = \{\mathbf{G}, \mathbf{F}, \mathbf{h}_t, \boldsymbol{\beta}_t\}$ can be obtained by maximizing the likelihood defined in Eqn. 16. We then apply the gradient based method L-BFGS [10] to update model parameters by using the following derivation formulas respectively:

$$\frac{\partial\mathcal{L}}{\partial\mathbf{F}} = -\sum_{i=1}^{N}\sum_{t=1}^{n_i}\left(\mathbf{s}_{i,t}^{(n_i)} - \mathbf{F}\mathbf{s}_{i,t-1}^{(n_i)} - \mathbf{G}\mathbf{x}_{i,t}\right)\mathbf{Q}_{i,t}^{-1}\mathbf{s}_{i,t-1}^{(n_i)} \quad (17)$$

$$\frac{\partial\mathcal{L}}{\partial\mathbf{G}} = -\sum_{i=1}^{N}\sum_{t=1}^{n_i}\left(\mathbf{s}_{i,t}^{(n_i)} - \mathbf{F}\mathbf{s}_{i,t-1}^{(n_i)} - \mathbf{G}\mathbf{x}_{i,t}\right)\mathbf{Q}_{i,t}^{-1}\mathbf{x}_{i,t} \quad (18)$$

$$\frac{\partial\mathcal{L}}{\partial\mathbf{h}_t} = \sum_{i=1}^{N}\sum_{t=1}^{n_i}\left(y_{i,t} - \sigma(\mathbf{h}_t^T\mathbf{s}_{i,t}^{(n_i)} + \boldsymbol{\beta}_t^T\mathbf{x}_{i,t})\right)\mathbf{s}_{i,t}^{(n_i)} \quad (19)$$

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\beta}_t} = \sum_{i=1}^{N}\sum_{t=1}^{n_i}\left(y_{i,t} - \sigma(\mathbf{h}_t^T\mathbf{s}_{i,t}^{(n_i)} + \boldsymbol{\beta}_t^T\mathbf{x}_{i,t})\right)\mathbf{x}_{i,t} \quad (20)$$

Initialization of the EM Algorithm: The initial value of parameters Φ should be chosen with care, otherwise the EM algorithm may not converge. In our experiment, the matrix $\mathbf{G}$ is initially set as the transform matrix resulted from principle component analysis (PCA) algorithm [7], and the matrix $\mathbf{F}$ is assigned to be an identity matrix.

4. EXPERIMENT

In order to evaluate the performance of our proposed model, we conducted an experiment on a real-life dataset.

4.1 Dataset

We use a data set collected from xuetangX [5], one of the largest MOOC platforms in China. This dataset was released for KDD CUP 2015 [6]. The dataset, as shown in Table 2, includes 79,186 students each of whom enrolled on at least one course among the whole set of 39 courses. Each enrollment is associated with a log of the student's activities including watching lecture videos, working on course's problems, accessing course's modules, and so on. Totally, there are 8,157,277 activity logs and the longest lifetime of enrollment is 5 weeks.

Table 2: Statistics of xuetangX dataset for the experiment

Item	Statistical description
# courses	39
# students	79,186
# enrollments	120,542
# activity logs	8,157,277
# longest lifetime of enrollment	5 weeks

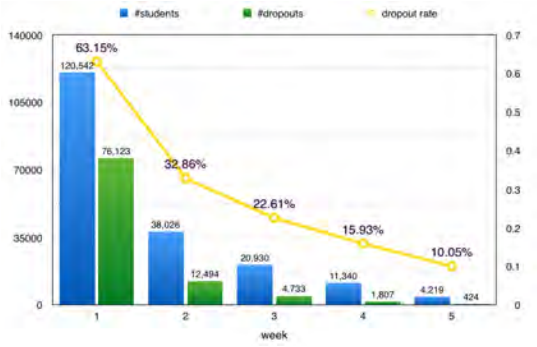


Figure 2: The number of students, number of dropouts, and the dropout rate in different weeks.

As shown in Figure 2, we observe that 76,123 students dropped out in the first week. Another observation is that the longer the student has engaged with the course, the less likely s/he quit the course. For example, the dropout rate of students who have engaged with the courses for 5 weeks is 10.05% vs. 63.15% for 1 week.

4.2 Evaluation Metrics

Due to the class imbalance phenomenon, we use Area Under the Receiver Operating Characteristics Curve (AUC) as the evaluation metric, as it is invariant to imbalance. Concretely, AUC measures how likely a classifier can correctly discriminate between positive and negative samples. An AUC of 1 indicates perfect discrimination whereas 0.5 corresponds to a classifier that guesses randomly.

[5]http://www.xuetangx.com
[6]http://www.kddcup2015.com

4.3 Compared Methods

We compared our model with related methods:

- Logistic Regression (LG) [14]: In this method, a logistic regression classifier is trained to make dropout prediction for each week. Specifically, for a student i in week t, his/her dropout probability is computed as the logistic function of the weighted sum of input features $\mathbf{x}_{i,t}$:

$$p(y_{i,t}|\mathbf{x}_{i,t}, \mathbf{w}_t) = \frac{1}{1+\exp(-y_{i,t}\mathbf{w}_t^T\mathbf{x}_{i,t})} \quad (21)$$

  where $\mathbf{w}_t = [w_{t1}, w_{t2}, \ldots, w_{tN}]^T$ is the weight vector to be learned. The objective function for week t is

$$\mathcal{L}(\mathbf{w}_t) = \sum_{i\in N_t} \log(1+\exp(-y_{i,t}\mathbf{w}_t^T\mathbf{x}_{i,t})) + \frac{\lambda_1}{2}||\mathbf{w}_t||^2 \quad (22)$$

  where N_t is the set of students who engage with the course in week t and $\lambda_1 > 0$ is the regularization parameter for $\mathbf{w}_t$.

- Simultaneously Smoothed Logistic Regression (LR-SIM) [6]: It extends the logistic regression by smoothing the predicted dropout probabilities across consecutive weeks. In this model, a regularization term is added into the objective function to minimize the difference of the predicted probabilities between two adjacent weeks, such as $\mathbf{w}_t^T\mathbf{x}_{i,t}$ and $\mathbf{w}_{t-1}^T\mathbf{x}_{i,t-1}$. A new feature space $\mathbf{x}'_{i,t}$ is introduced, which has $T \times N$ dimensions (T is the total number of weeks), with the t-th component having N features corresponding to the features in the original feature space $\mathbf{x}_{i,t}$ for week t, and other $T-1$ components corresponding to zeroes. Then, a single weight vector $\mathbf{w}$ is introduced, which also has $T \times N$ dimensions corresponding to $\mathbf{x}'_{i,t}$. The final objective function is defined as:

$$\mathcal{L}(\mathbf{w}) = \sum_{i\in N_t}\sum_{t=1}^{n_i} \log\left(1+\exp(-y_{i,t}\mathbf{w}^T\mathbf{x}'_{i,t})\right) + \frac{\lambda_1}{2}||\mathbf{w}||^2 + \lambda_2 \sum_{t=2}^{T}\sum_{i\in N_{t,t-1}} ||\mathbf{w}^T\mathbf{x}'_{i,t} - \mathbf{w}^T\mathbf{x}'_{i,t-1}||^2 \quad (23)$$

  where $N_{t,t-1}$ is the set of students who engage with the course in both weeks t and $t-1$, and $\lambda_2 > 0$ is the regularization parameter for the difference of the resulted dropout probabilities between two adjacent weeks.

- RNN with Long Short-Term Memory Cell (LSTM) [12]: It uses a recurrent neural network (RNN) model with long short-term memory (LSTM) architecture to train a sequence classifier model that produces temporal prediction. Similar to our proposed model, given the student's week-by-week features and dropout labels $\{(\mathbf{x}_{i,t}, y_{i,t}), 1 \le t \le n_i\}$, the LSTM model is applied to estimate the student state, which can then be used to predict the student's future actions.

Note that we did not compare with Hidden Markov Model (HMM) based method [2] because it can be treated as a special case of RNN by representing student state as discrete variable. For all the compared models, we used the same set of features as input (see Table 1).

4.4 Results and Discussion

The main hyperparameter to determine the NSSM model's performance is the dimensionality of student state K (see Eqn. 1). We compared the performance of NSSM in terms of AUC with varying dimension of latent state K, and observed that the optimal value of K in most cases is 12. Therefore, in our experiment, we set K as 12 to train the NSSM model.

4.4.1 Single Course

In this setting, we trained a separate model for each course. To get sufficient data for training, we only consider the popular courses that include more than 5,000 students. After filtering, 6 popular courses are used in this experiment. As students may enroll in a course at different time steps, we select 70% students who enrolled in the course in early period as the training data, and remaining 30% students as the testing data.

	LR	LR-SIM	LSTM	NSSM
Week 1	0.812	0.886	0.891	**0.900**
Week 2	0.819	0.876	0.887	**0.891**
Week 3	0.807	0.854	0.861	**0.870**
Week 4	0.768	0.778	0.786	**0.796**
Week 5	0.673	0.679	0.689	**0.702**

Table 3: Performance comparison of LR, LR-SIM, LSTM and NSSM in terms of average AUC on 6 popular courses.

Table 3 presents the average AUC scores across weeks by testing different models. The results indicate that the models that consider dependence between consecutive weeks, such as LR-SIM, LSTM and NSSM, achieve higher AUC score than the baseline LR model without this consideration. For example, for the first week, the AUC score of NSSM is 0.9, which is 10.8% improvement relative to that of LR model. Furthermore, we can see that the methods that model the student's states over time (i.e., LSTM and NSSM) achieve higher AUC than LR and LR-SIM in most cases. More notably, our proposed model NSSM performs consistently better than LSTM, suggesting that the student states estimated by NSSM is more predictive than those by LSTM. We can also observe that the accuracy during early weeks is higher than that of later weeks by most of models. This implies that the dropout prediction task may become harder with increasing lifetime of engagement, as there might be various hidden reasons that cause a student to quit the course.

4.4.2 Across Courses

In this setting, we are interested in evaluating whether the proposed model trained on some courses can serve other courses as well, for which we randomly select 70% courses for training and remaining 30% for testing. In this experiment, we use all of the student data from the training courses to train the model.

Table 4 shows the performance comparison. Same conclusions can be made as in the previous Section 4.4.1. Specifically, from this table, we can observe that our proposed model NSSM still outperforms the other models (e.g., LR, LR-SIM and LSTM) across different weeks. For example,

	LR	LR-SIM	LSTM	NSSM
Week 1	0.835	0.933	0.936	**0.936**
Week 2	0.911	0.915	0.915	**0.919**
Week 3	0.868	0.872	0.867	**0.871**
Week 4	0.782	0.784	0.785	**0.789**
Week 5	0.655	0.662	0.673	**0.686**

Table 4: Performance comparison of LR, LR-SIM, LSTM and NSSM in terms of AUC on new courses across weeks.

for the first week, the AUC score of NSSM is 0.686, which is 12% improvement relative to that of LR model. Furthermore, we can see that the improvement from NSSM with regard to LSTM is slight, and the relative improvement during later weeks is larger than that of early weeks (e.g., +5.1% during week 4 vs +4.4% during week 2). This observation implies that the NSSM has the potential to make better dropout predictions for students who have longer lifetime of engagement than LSTM. In addition, as these results are predictions made for students from new courses, we can conclude that our proposed model is capable of making better dropout prediction in new courses, in comparison with other models.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have focused on identifying at-risk students in online courses by making dropout prediction. We particularly take advantage of nonlinear state space model (NSSM) because it can discover a student's latent state to characterize the student's intention to perform certain activities. We conducted experiment on a real-world dataset, which demonstrates that our proposed model achieves higher prediction accuracy than related methods. We also showed that the NSSM model trained on data from some courses can make dropout prediction for students in new courses.

However, because the extended Kalman filter and smoother we used in this paper may not be an optimal parameter estimator, the difference between NSSM and LSTM is slight. Therefore, in the future, we will exploit other advanced algorithms (e.g., Unscented Kalman filter) to estimate the parameters in our nonlinear state space model. For the second future direction, as the experiment presented in this paper is limited to xuetangX dataset, we plan to evaluate our proposed model on datasets collected from other MOOC platforms, such as Edx and Coursera.

6. ACKNOWLEDGMENTS

This research work was supported by Hong Kong GRF ECS/HKBU211912 and partially supported by ITF ITS/271/14FX.

7. REFERENCES

[1] H. J. Andrew. *Stochastic Processes and Filtering Theory*. Academic Press, Inc., New York and London, 1970.

[2] G. Balakrishnan. Predicting student retention in massive open online courses using hidden markov models. Master's thesis, EECS Department, University of California, Berkeley, May 2013.

[3] M. Y. Byron, K. V. Shenoy, and M. Sahani. Derivation of extended kalman filtering and smoothing equations. Technical report, 2004.

[4] D. S. Chaplot, E. Rhim, and J. Kim. Predicting student attrition in moocs using sentiment analysis and neural networks. In *Proc. of the 2015 AIED Workshop on Intelligent Support for Learning in Groups*, pages 7–12, 2015.

[5] S. Halawa, D. Greene, and J. Mitchell. Dropout prediction in moocs using learner activity features. *Experiences and Best Practices in and around MOOCs*, 7:7–16, 2014.

[6] J. He, J. Bailey, B. I. P. Rubinstein, and R. Zhang. Identifying at-risk students in massive open online courses. In *Proc. of the 29th AAAI Conference on Artificial Intelligence*, pages 1749–1755, 2015.

[7] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417, 1933.

[8] K. Jordan. Mooc completion rates: The data. *Availabe at: http://www.katyjordan.com/MOOCproject.html. [Accessed: 04/02/2016]*, 2016.

[9] M. Kloft, F. Stiehler, Z. Zheng, and N. Pinkwart. Predicting mooc dropout over weeks using machine learning methods. In *Proc. of the 2014 EMNLP Workshop on Analysis of Large Scale Social Interaction in MOOCs*, pages 60–65, 2014.

[10] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

[11] W. Mader, Y. Linke, M. Mader, L. Sommerlade, J. Timmer, and B. Schelter. A numerically efficient implementation of the expectation maximization algorithm for state space models. *Applied Mathematics and Computation*, 241:222–232, 2014.

[12] F. Mi and D.-Y. Yeung. Temporal models for predicting student dropout in massive open online courses. In *Proc. of the 2015 ICDM Workshop on Data Mining for Educational Assessment and Feedback*, pages 256–263, November 2015.

[13] A. Ramesh, D. Goldwasser, B. Huang, H. Daumé, III, and L. Getoor. Learning latent engagement patterns of students in online courses. In *Proc. of the 28th AAAI Conference on Artificial Intelligence*, pages 1272–1278, 2014.

[14] C. Taylor, K. Veeramachaneni, and U.-M. O'Reilly. Likely to stop? Predicting stopout in massive open online courses. *arXiv preprint arXiv:1408.3382*, 2014.

[15] D. Yang, T. Sinha, D. Adamson, and C. P. Rose. "Turn on, Tune in, Drop out": Anticipating student dropouts in massive open online courses. In *Proc. of the 2013 NIPS Data-Driven Education Workshop*, volume 11, 2013.

Transactivity as a Predictor of Future Collaborative Knowledge Integration in Team-Based Learning in Online Courses

Miaomiao Wen, Keith Maki, Xu Wang, Steven P Dow, James Herbsleb and Carolyn Rose
Carnegie Mellon University
Pittsburgh, USA
{mwen,kmaki,xuwang,spdow,jdh,cprose}@cs.cmu.edu

ABSTRACT

To create a satisfying social learning experience, an emerging challenge in educational data mining is to automatically assign students into effective learning teams. In this paper, we utilize discourse data mining as the foundation for an online team-formation procedure. The procedure features a deliberation process prior to team assignment, where participants hold discussions both to prepare for the collaboration task and provide indicators that are then used during automated team assignment. We automatically assign teams in a way that maximizes average observed pairwise transactivity exchange within teams, whereas in a control condition, teams are assigned randomly. We validate our team-formation procedure in a crowdsourced online environment that enables effective isolation of variables, namely Amazon's Mechanical Turk. We compare group knowledge integration outcomes between the two team assignment conditions. Our results demonstrate that transactivity-based team assignment is associated with significantly greater knowledge integration ($p < .05$, effect size 3 standard deviations).

1. INTRODUCTION

Although there are typically thousands of students in a Massive Open Online Course (MOOC), social isolation is still the norm in the current generation of MOOCs. However, there is evidence that many students would prefer to have more social engagement in that context. Recent research shows that a quarter of learners want to meet new people in their courses; and another 20% of learners in typical MOOCs want to take their courses with friends or colleagues [17]. To satisfy learners' social needs, there is growing interest in enabling group learning in MOOC learning contexts. Recent emerging platforms like NovoEd[1] and cMOOCs are designed with team-based learning or social interaction at center stage. Additionally, many recent xMOOCs are adopting team-based learning features (e.g., in EdX[2]). There is accumulating evidence that social interaction is associated with enhanced commitment to the course [11], which has the potential to address one of MOOC critics' biggest concerns, namely high attrition rates [18]. However, how to automatically assign students to effective MOOC learning groups is still an open question [12, 25, 20]. Methods for mining educational data have been used to optimize instruction or feedback for individuals [21]. In this paper we explore how a form of educational data mining (namely, mining of discussion behavior) can be used to optimize the experience of collaborative learners through the support of effective team formation.

Algorithms for group assignment typically bring together students based on learning style, personality or demographic information. For team assignments based on such algorithms, student information must be collected and then provided to the algorithm [9]. Because of the paucity of available student personal information in MOOCs, designing a team-formation process that relies on mining of discussion data to fill in missing information would be a valuable contribution. Moreover, research identifying valuable evidence for effective team formation is needed since recent work shows that forming teams based on typical demographic features, e.g. gender and time zone, does not significantly improve teams' engagement and success in MOOCs [25]. In an online interaction, demographic information about learners is only relevant to the extent that it influences how those students come across and interact with others. Thus, observation of behavior and interaction between students may be a better source of insight for assigning students to groups in which they will function well as a team. This provides an excellent opportunity for data mining technology to make a contribution in support of valued learning processes. The alternative to automated assignment is self-selected teams. When a student population is large, which is usually the case in MOOCs, it is difficult for students to navigate through a list of students or teams to find a team that fits. Previous work has shown that many self-selected teams fail in team-based MOOCs [23]. As an alternative to both of these approaches, we design a practical group-formation procedure through which participants are organized into small groups

[1] https://novoed.com

[2] https://courses.edx.org/courses/course-v1:McGillX+GROOCx+T3_2015, https://www.edx.org/course/medicinal-chemistry-molecular-basis-drug-davidsonx-d001x-1

based on the data mined from their participation processes in the course. This procedure uses a deliberation process, where participants hold discussions in preparation for the collaboration task; teams are then automatically assigned based on features of their interaction during deliberation.

In recent years there has been increasing interest in mining discourse data for insights into learning processes [7], for understanding factors associated with attrition in MOOCs [16], and for building models to trigger dynamic support for collaborative learning [11]. In this paper, we mine students' collaborative process to collect information for automatic team assignment. In particular, we automatically identify an important property of discourse, transactivity, from students' discussion. Transactivity is known to be higher within groups where there is mutual respect [5] and a desire to build common ground [14]. Previous studies showed that high transactivity groups are associated with higher learning [22], higher knowledge transfer [13], and better problem solving [5]. Prior work has demonstrated success at automatic detection of transactivity and relevant discussion constructs [14]. Because of the social underpinnings of transactivity, it is reasonable to hypothesize that automated detection of transactivity could form the basis for an automated group assignment procedure in online learning contexts. In this paper, we combine text-mining and algorithm-based team formation; We study whether by grouping individuals with a history of engaging in more transactive communication during a pre-collaboration deliberation can help them achieve more effective collaboration in their teams. Simply stated, our research question is:

Can evidence of transactive discussions during deliberation inform the formation of more successful teams?

As a step towards effective team-based learning in MOOCs, in this paper, we explore the team-formation process in an experimental study conducted in an online setting that enables effective isolation of variables, namely Amazon's Mechanical Turk (MTurk). While crowd workers likely have different motivations from MOOC students, their remote individual work setting without peer contact resembles today's MOOC setting where most students learn in isolation [6]. This allows us to test the causal connection between variables in order to identify principles that later we will test in an actual MOOC. A similar approach was taken in prior work to inform design of MOOC interventions for online group learning [6]. We designed a collaborative knowledge integration task where participants work together on writing an energy proposal for a city. This knowledge integration task is modeled after ones used in earlier collaborative learning studies [4]. The participants in our study will be referred to as students throughout the paper.

2. METHODS

Our experimental study is designed as a validation of a team-formation paradigm. In this paradigm, we attempt to offer teams a running start in their collaboration work by starting them with individual work, which they then discuss as a community. In addition to providing the basis for assignment to teams, the community engagement prior to team formation provides students with a breadth of exposure to different perspectives relevant to the group work. Based on the interactions displayed during this community discussion, students are automatically assigned to teams. The students then enter their teams for the bulk of their group work. We test a transactivity-maximization team-formation method. Instead of grouping students high in transactivity into teams and students low in transactivity together, the team assignment algorithm maximizes the average amount of transactive communication within all the teams through a constraint satisfaction algorithm.

2.1 Experimental Paradigm

2.1.1 Collaboration Task Description

For the team task, we designed a highly-interdependent collaboration task that requires negotiation in order to create a context in which effective group collaboration would be necessary for task success. The task is comparable to a course project where a student team writes a proposal collaboratively. We used a Jigsaw paradigm, which has been demonstrated as an effective way to achieve a positive group composition and is associated with positive group outcomes [4]. In a Jigsaw task, each student is given a portion of the knowledge or resources needed to solve the problem, but no one has enough to complete the task alone. Following the Jigsaw paradigm, each member of the team was given special knowledge of one of the four energy sources, and was instructed to represent the values associated with their energy source in contrast to the rest, e.g. coal energy was paired with an economical energy perspective. The team collaborative task was to select a single energy plan and write a proposal arguing in favor of the group decision with respect to the associated trade-offs, meaning team members needed to negotiate a prioritization among the city requirements with respect to the advantages and disadvantages they were cumulatively aware of. The set of potential energy plans was constructed to reflect different trade-offs among the requirements, with no plan satisfying all of them perfectly. This ambiguity created an opportunity for intensive exchange of perspectives. The collaboration task is shown in Figure 1.

2.1.2 Experimental Procedure

We designed a four-step process for the task:

Step 1: Preparation. In this step, each student was asked to provide a nickname, which would be used in the deliberation and collaboration phases. To prepare for the Jigsaw task, each student was randomly assigned to read an instructional article about the pros and cons of a single energy source. Each article was approximately 500 words, and covered one of four energy sources (coal, wind, nuclear, and hydro power). To strengthen their learning and prepare them for the proposal writing, we asked them to complete a quiz reinforcing the content of their assigned article. The quiz consisted of 8 single-choice questions, and feedback including correct answers and explanations was provided along with the quiz.

Step 2: Pre-task. In this step, we asked each student to write a proposal to recommend one of the four energy sources (coal, wind, nuclear, and hydro power) for a city given five requirements, e.g. "The city prefers a stable energy". After each student finished this step, their proposal was automatically posted in a forum as the start of a thread with the title "[Nickname]'s Proposal".

In this final step, you will work together with other Turkers to recommend a way of distributing resources across energy types for the administration of City B. City B requires 12,000,000 MWh electricity a year from four types of energy sources: coal power, wind power, nuclear power and hydro power. We have provided 4 different plans to choose from, each of which emphasizes one energy source as primary. Your team needs to negotiate which plan is the best way of meeting your assigned goals, given the city's requirements and information below.
City B's requirements and information:
1. City B has a tight yearly energy budget of $900,000K. Coal power costs $40/MWh. Nuclear power costs $100/MWh. Wind power costs $70/MWh. Hydro power costs $100/MWh.
2. The city is concerned with chemical waste. If the main energy source releases toxic chemical waste, there is a waste disposal cost of $2/MWh.
3. The city is a famous tourist city for its natural bird and fish habitats.
4. The city is trying to reduce greenhouse gas emissions. If the main energy source releases greenhouse gases, there will be a "Carbon tax" of $10/MWh of electricity.
5. The city has several large hospitals that need a stable and reliable energy source.
6. The city prefers renewable energy. If renewable energies generate more than 30% of the electricity, there will be a renewable tax credit of $1/MWh for the electricity that is generated by renewable energies.
7. The city prefers energy sources whose cost is stable.
8. The city is concerned with water pollution.

	Energy Plan				Cost	Waste disposal cost	Carbon tax	Renewable tax credit	Total
	Coal	Wind	Nuclear	Hydro					
Plan 1	40%	20%	20%	20%	$840,000K	$14,400K	$48,000K	$9,600K	$892,800K
Plan 2	20%	40%	20%	20%	$912,000K	$0	$0	$11,000K	$901,000K
Plan 3	20%	20%	40%	20%	$984,000K	$14,400K	$0	$9,600K	$988,800K
Plan 4	20%	20%	20%	40%	$984,000K	$0	$0	$11,000K	$973,600K

Figure 1: This figure displays the collaborative task as it was presented to the students. In addition to the task statement, they had a chat interface and a shared document space to work in.

Step 3: Deliberation. In this step, students joined a threaded forum discussion akin to those available in many online environments. Each proposal written by the students in the Pre-task (Step 2) was displayed for students to read and comment on. Each student was required to write at least five replies to the proposals posted by the other students. To encourage the students to discuss transactively, the task instruction for this step included the request to, when replying to a post, "elaborate, build upon, question or argue against the ideas presented in that post, drawing from the argumentation in your own proposal where appropriate."

Step 4: Collaboration. In the collaboration step, team members in a group were first gathered for synchronous interaction and then directed to a shared document space to write a proposal together to recommend one of four suggested energy plans based on a city's eight requirements. Students in the same team were able to see each other's edits in real time, and were able to communicate with each other using a synchronous chat utility on the right sidebar. The collaborative task was designed to contain richer information than the individual proposal writing task in Step 2.

2.1.3 Outcome Measures

We evaluated team success using two types of outcomes, namely objective success through quantitative task performance (i.e., the quality of the integrated proposal, which indicates collaborative knowledge integration [3]) and process measures, as well as subjective success through a group satisfaction survey. The quantitative task performance measure was an evaluation of the quality of the proposal produced by the team. The goal of evaluating the team knowledge integration process is to distinguish instances when students are making statements based on reasoning from simply repeating what they have read. In particular, the scoring rubric defined how to identify the following elements for a proposal: (1) Which requirements were considered; (2) Which comparisons or trade-offs were made; (3) Which additional valid desiderata were considered beyond stated requirements; (4) Which incorrect statements were made about requirements. Positive points were awarded to each proposal for correct requirements considered, comparisons made, and additional valid desiderata. Negative points were awarded for incorrect statements. We measured *Team Knowledge Integration* by the total points assigned to the team proposal, i.e. team proposal score. Two PhD students who were blind to the conditions applied the rubric to five proposals (a total of 78 sentences) and the inter-rater reliability was good (Kappa = 0.74). The two raters then coded all the proposals.

We used the *length of chat discussion* during teamwork as a measure of team process in the Collaboration step. On average the longer discussions referred to more substantive issues.

Group Experience Satisfaction was measured using a four item group experience survey administered to each student after the Collaboration step. The survey was based on items used in prior work [19, 6]. In particular, the survey instrument included items related to:

- Satisfaction with team experience.
- Satisfaction with proposal quality.
- Satisfaction with the group communication.
- Perceived learning through the group experience.

Each of the items was measured on a 7-point Likert scale.

2.1.4 Control Variables

Intuitively, students who display more effort in the Pre-task might perform better in the collaboration task, so that level of effort is an important control variable. We used each student's individual Pre-task proposal length as a control variable for Individual Performance. Analogously, we used each group's average group member Pre-task proposal length as a control variable for the group knowledge integration analyses.

2.1.5 Transactivity Annotation, Prediction, and Measurement

To enable us to use counts of transactive contributions as evidence to inform an automated group assignment procedure, we needed to automatically judge whether a reply post in the Deliberation step was transactive or not using machine learning. A transactive contribution displays the author's reasoning and connects that reasoning to material communicated earlier. Two example posts illustrating the contrast are shown below:

- Transactive
  "Nuclear energy, as it is efficient, it is not sustainable. Also, think of the disaster probabilities".

- Non-transactive
  "I agree that nuclear power would be the best solution".

Using a validated and reliable coding manual for transactivity from prior work [14], an annotator previously trained to apply that coding manual annotated 426 reply posts collected in pilot studies we conducted in preparation for the studies reported in this paper. Each of those posts was annotated as either "transactive" or "non-transactive". 70% of them were transactive.

Automatic annotation of transactivity has been reported in the Computer Supported Collaborative Learning literature. For example, researchers have applied machine learning using text, such as chat data [15] and transcripts of whole group discussions [2]. We trained a Logistic Regression model with L2 regularization using a set of features consisting of single word features (i.e., unigrams) as well as a feature indicating the post length [10]. We evaluated our classifier with a 10-fold cross validation and achieved an accuracy of 0.843 and a 0.615 Kappa. Given the adequate performance of the model, we used it to predict whether each reply post in the Deliberation step was transactive or not.

To measure *the amount of transactive communication* between two students in the Deliberation step, we counted the number of times a pair of their posts in a same discussion thread were transactive; or one of them was a thread starter and the other student's reply was transactive.

2.2 Transactivity Maximization Grouping

The Transactivity Maximization teams were formed so that the average amount of transactive discussion observed in the Deliberation step among the team members in the team was maximized. A Minimal Cost Max Network Flow algorithm was used to perform this constraint satisfaction process [1]. This network flow algorithm tackles resource allocation problems with constraints. In our case, we need to satisfy the Jigsaw constraint. At the same time, the minimal cost part of the algorithm maximized the transactive communication that was observed among the group members during the Deliberation step. The algorithm finds an approximately optimal grouping within $O(N^3)$ (N = number of students) time complexity. A brute force search algorithm, which has an $O(N!)$ time complexity, would take too long to finish in real time.

Our algorithm can achieve an approximately optimal solution in an admissible time. Instead of maximizing the pair-wise accumulated transitivity post count, we approximate the solution by maximizing the accumulated transitivity post count between two adjacent pairs of users. The algorithm can be generalized to form teams of any size. In our experiment, the team size is 4. We build a directed weighted graph based on students' discussion network. Then we use the successive shortest path algorithm to find a suboptimal, but nevertheless substantially better than random grouping [1]. The algorithm greedily finds a flow with minimum cost until there is no remaining flow in the network, as outlined in Algorithm 1.

Algorithm 1 Successive Shortest Paths for Minimum Cost Max Flow

$f(v_1, v_2) \leftarrow 0 \ \forall (v_1, v_2) \in E$
$E' \leftarrow a(v_1, v_2) \ \forall (v_1, v_2) \in E$
while $\exists \Pi^* \in G' = (V, E')$
s.t. Π^* *a minimum cost path from* S *to* D **do**
  for each $(v_1, v_2) \in \Pi^*$
  if $f(v_1, v_2) > 0$ **then**
    $f(v_1, v_2) \leftarrow 0$
    remove $-a(v_2, v_1)$ from E'
    add $a(v_1, v2)$ to E'
  else
    $f(v_1, v_2) \leftarrow 1$
    remove $a(v_1, v_2)$ from E'
    add $-a(v_2, v1)$ to E'
  end
end

2.2.1 Experimental Manipulation

In our study, students participated in a deliberative discussion as a community in a threaded discussion forum prior to being assigned to teams automatically. We investigated how the nature of the experience in that context may contribute to the success of the teams. We made use of a Jigsaw paradigm in the team assignment of teams in both the experimental and control conditions. In the experimental condition, which we termed the Transactivity Maximization condition, we additionally applied a constraint that preferred to maximize the extent to which students assigned to the same team had participated in automatically detected transactive exchanges in the deliberation. In the control condition, which we termed the Random condition, apart from enforcing the Jigsaw constraint, teams were formed by random assignment. In this way we tested the hypothesis that observed transactivity is an indicator of potential for effec-

tive team collaboration. We ran the study in 10 separate batches, with 5 batches in each condition. In each batch, all the students in that batch were assigned to teams using the Random strategy or all the students were assigned to teams using the Transactivity Maximization strategy. The average level of amount of transactivity during the deliberation stage was not significantly different between batches. Thus we can test if the team-formation method can predict future collaborative knowledge integration. All the steps and instructions of the task were identical for the two conditions, as described in 2.1.2.

2.3 Participants

Participants were recruited on MTurk with the qualifications of having a 95% acceptance rate on 1,000 tasks or more. Each student was only allowed to participate once. A total of 246 students participated in the experiment, the students who were not assigned into groups or did not complete the group satisfaction survey were excluded from our analysis. The experiment lasted on average 35.9 minutes. We included only teams of 4 students in our analysis. There were in total 27 Transactive Maximization teams and 27 Random teams, with no significant difference in attrition between conditions ($\chi^2(1) = 1.46$, $p = 0.23$). The dropout rate of students in Random groups was 27%. The dropout rate of students in Transactivity Maximization groups was 19%.

3. RESULTS

As a manipulation check, we compared the average amount of transactivity observed among teammates during the deliberation between the two conditions using a t-test. The groups in the Transactive Maximization condition (M = 12.85, SD = 1.34)[3] were observed to have had significantly more transactive exchanges during the deliberation than those in the Random condition (M = 7.00, SD = 1.52) ($p < 0.01$), with an effect size of 3.85 standard deviations, demonstrating that the maximization was successful in manipulating the average experienced transactive exchange within teams between conditions.

Teams that experienced greater transactivity during deliberation demonstrate better team knowledge integration.
To assess whether the Transactivity Maximization condition resulted in more effective teams, we tested for a difference between group-formation conditions on Team Knowledge Integration. We built an ANOVA model with Grouping Criteria (Random, Transactivity Maximization) as the independent variable and Team Knowledge Integration as the dependent variable. Average team member Pre-task proposal length was again the covariate. There was a significant main effect of Grouping Criteria (F(1,52) = 6.13, $p < 0.05$) on Team Knowledge Integration such that Transactivity Maximization teams (M = 11.74, SD = 0.67) demonstrated significantly better performance than the Random groups (M = 9.37, SD = 0.67) ($p < 0.05$), with an effect size of 3.54 standard deviations, which is a large effect. Effect size is measured in terms of Cohen's d.

Across the two conditions, observed transactive communication during deliberation was significantly correlated with Team Knowledge Integration (r = 0.26, $p < 0.05$). This also indicated teams that experienced more transactive communication during deliberation demonstrated better Team Knowledge Integration.

Teams that experienced greater transactivity during deliberation demonstrate more intensive interaction within their teams.
In the experiment, students were assigned to teams based on observed transactive communication during the deliberation step. Assuming that individuals that were able to engage in positive collaborative behaviors together during the deliberation would continue to do so once in their teams, we would expect to see evidence of this reflected in their observed team process. Group processes have been demonstrated to be strongly related to group outcomes in face-to-face problem solving settings [24]. Thus, we should consider evidence of a positive effect on group processes as an additional positive outcome of the experimental manipulation.

In order to test whether such an effect occurred, we built an ANOVA model with Grouping Criteria (Random, Transactivity Maximization) as the independent variable and length of chat discussion during teamwork as the dependent variable. There was a significant effect of Grouping Criteria on length of discussion (F(1,45) = 9.26, $p < 0.005$). Random groups (M = 20.00, SD = 3.58) demonstrated significantly shorter discussions than Transactive Maximization groups (M = 34.52, SD = 3.16), with an effect size of 4.06 standard deviations.

Survey results
For each of the four aspects of the group experience survey, we built an ANOVA model with Grouping Criteria (Random, Transactivity Maximization) as the independent variable and the survey outcome as the dependent variable. Team ID and assigned energy condition (Coal, Wind, Hydro, Nuclear) were included as control variables nested within condition. There were no significant effects on Satisfaction with team experience or with proposal quality. However, there was a significant effect of condition on Satisfaction with communication within the group (F(1,112) = 4.83, $p < 0.05$), such that students in the Random teams (M = 5.12, SD = 1.7) rated the communication significantly lower than those in the Transactivity Maximization teams (M = 5.69, SD = 1.51), with effect size 0.38 standard deviations. Additionally, there was a marginal effect of condition on Perceived learning (F(1,112) = 2.72, $p = 0.1$), such that students in the Random teams (M = 5.25, SD = 1.42) rated the perceived benefit to their understanding they received from the group work lower than students in the Transactivity Maximization teams (M = 5.55, SD = 1.27), with effect size 0.21 standard deviations. Thus, with respect to subjective experience, we see advantages for the Transactivity Maximization condition, but the results are weaker than those observed for the objective measures. Nevertheless, these results are consistent with prior work where objectively measured learning benefits are observed in high transactivity teams [8].

4. DISCUSSION

In this paper we presented an experiment to address our research question regarding the extent to which benefit could be achieved by selecting teams based on evidence of trans-

[3]SD is short for standard deviation in this paper.

active exchange observed during the deliberation. We designed an automatic team-formation process that combines discourse data mining and algorithm-based team formation. Here we found that teams formed such that observed transactive interactions between team members in the deliberation was maximized displayed objectively better knowledge integration than teams assigned randomly. On subjective measures we see a significant positive impact of transactivity maximization on perceived communication quality and a marginal impact on perceived enhanced understanding, both of which are consistent with what we would expect from the literature on transactivity where high transactivity teams have been demonstrated to produce higher quality outcomes and greater learning [22]. These results provide positive evidence in favor of a design for a team-formation strategy in two stages: Individuals first participate in a pre-teamwork deliberation activity where they explore the space of issues in a context that provides beneficial exposure to a wide range of perspectives. Individuals are then grouped automatically through a transactivity detection and maximization procedure that uses communication patterns arising naturally from community processes to inform group formation with an aim for successful collaboration.

This research was supported in part by funding from Google and the Gates foundation.

5. REFERENCES

[1] R. K. Ahuja and J. B. Orlin. A fast and simple algorithm for the maximum flow problem. *Operations Research*, 37(5):748–759, 1989.

[2] H. Ai, R. Kumar, D. Nguyen, A. Nagasunder, and C. P. Rosé. Exploring the effectiveness of social capabilities and goal alignment in computer supported collaborative learning. In *Intelligent Tutoring Systems*, pages 134–143. Springer, 2010.

[3] M. Alavi and A. Tiwana. Knowledge integration in virtual teams: The potential role of kms. *Journal of the American Society for Information Science and Technology*, 53(12):1029–1037, 2002.

[4] E. Aronson. *The jigsaw classroom.* Sage, 1978.

[5] M. Azmitia and R. Montgomery. Friendship, transactive dialogues, and the development of scientific reasoning. *Social development*, 2(3):202–221, 1993.

[6] D. Coetzee, S. Lim, A. Fox, B. Hartmann, and M. A. Hearst. Structuring interactions for large-scale synchronous peer learning. In *CSCW*, 2015.

[7] M. Dascalu, S. Trausan-Matu, D. S. McNamara, and P. Dessus. Readerbench: Automated evaluation of collaboration based on cohesion and dialogism. *International Journal of Computer-Supported Collaborative Learning*, 10(4):395–423, 2015.

[8] R. De Lisi and S. L. Golbeck. Implications of piagetian theory for peer learning. 1999.

[9] R. Decker. Management team formation for large scale simulations. *Developments in Business Simulation and Experiential Learning*, 22, 1995.

[10] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[11] O. Ferschke, I. Howley, G. Tomar, D. Yang, and C. Rose. Fostering discussion across communication media in massive open online courses. In *CSCL*, 2015.

[12] K. Ghadiri, M. H. Qayoumi, E. Junn, P. Hsu, and S. Sujitparapitaya. The transformative potential of blended learning using mit edx 6.002 x online mooc content combined with student team-based learning in class. *environment*, 8:14, 2013.

[13] G. Gweon. *Assessment and support of the idea co-construction process that influences collaboration.* PhD thesis, Carnegie Mellon University, 2012.

[14] G. Gweon, M. Jain, J. McDonough, B. Raj, and C. P. Rosé. Measuring prevalence of other-oriented transactive contributions using an automated measure of speech style accommodation. *International Journal of Computer-Supported Collaborative Learning*, 8(2):245–265, 2013.

[15] M. Joshi and C. P. Rosé. Using transactivity in conversation for summarization of educational dialogue. In *SLaTE*, pages 53–56, 2007.

[16] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, pages 170–179. ACM, 2013.

[17] R. F. Kizilcec and E. Schneider. Motivation as a lens to understand online learners: Toward data-driven design with the olei scale. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 22(2):6, 2015.

[18] C. Kulkarni, J. Cambre, Y. Kotturi, M. S. Bernstein, and S. Klemmer. Talkabout: Making distance matter with small groups in massive classes. In *CSCW*, 2015.

[19] I. Lykourentzou, A. Antoniou, and Y. Naudet. Matching or crashing? personality-based team formation in crowdsourcing environments. *arXiv preprint arXiv:1501.06313*, 2015.

[20] S. MacNeil, C. Latulipe, B. Long, and A. Yadav. Exploring lightweight teams in a distributed learning environment. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, pages 193–198. ACM, 2016.

[21] C. Piech, J. Huang, A. Nguyen, M. Phulsuksombati, M. Sahami, and L. Guibas. Learning program embeddings to propagate feedback on student code. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1093–1102, 2015.

[22] S. D. Teasley, F. Fischer, A. Weinberger, K. Stegmann, P. Dillenbourg, M. Kapur, and M. Chi. Cognitive convergence in collaborative learning. In *International conference for the learning sciences*, pages 360–367, 2008.

[23] M. Wen, D. Yang, and C. P. Rosé. Virtual teams in massive open online courses. *Proceedings of Artificial Intelligence in Education*, 2015.

[24] A. W. Woolley, C. F. Chabris, A. Pentland, N. Hashmi, and T. W. Malone. Evidence for a collective intelligence factor in the performance of human groups. *science*, 330(6004):686–688, 2010.

[25] Z. Zheng, T. Vogelsang, and N. Pinkwart. The impact of small learning group composition on student engagement and success in a mooc. *Proceedings of Educational Data Mining*, 7, 2015.

Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation

Kevin H. Wilson[*], Yan Karklin[*], Bojian Han[†], Chaitanya Ekanadham[*]
[*]Knewton, Inc. New York, NY [†]Carnegie Mellon University. Pittsburgh, PA
{kevin,yan,chaitu}@knewton.com bojianh@andrew.cmu.edu

ABSTRACT

Estimating student proficiency is an important task for computer based learning systems. We compare a family of IRT-based proficiency estimation methods to Deep Knowledge Tracing (DKT), a recently proposed recurrent neural network model with promising initial results. We evaluate how well each model predicts a student's future response given previous responses using two publicly available and one proprietary data set. We find that IRT-based methods consistently matched or outperformed DKT across all data sets at the finest level of content granularity that was tractable for them to be trained on. A hierarchical extension of IRT that captured item grouping structure performed best overall. When data sets included non-trivial autocorrelations in student response patterns, a temporal extension of IRT improved performance over standard IRT while the RNN-based method did not. We conclude that IRT-based models provide a simpler, better-performing alternative to existing RNN-based models of student interaction data while also affording more interpretability and guarantees due to their formulation as Bayesian probabilistic models.

Acknowledgements

Many thanks to Siddharth Reddy, David Kuntz, Kyle Hausmann, and Celia Alicata for discussions of this work and help editing the manuscript.

Keywords

Item Response Theory, Recurrent Neural Nets, Bayesian Models of Student Performance, Deep Knowledge Tracing

1. INTRODUCTION

A key challenge for computer-based learning systems is to estimate a student's proficiency based on her previous interactions with the system. Accurate estimation of proficiency enables more efficient diagnosis and remediation of her weaknesses and more effective advancement of her knowledge frontier. Proficiency estimates can also provide the student or teacher with actionable information to improve student outcomes when reported as analytics [21].

Two classical families of methods for estimating proficiency are Item Response Theory (IRT) [8, 13] and Bayesian Knowledge Tracing (BKT) [2]. IRT essentially amounts to structured logistic regression (see Section 2.1), estimating latent quantities corresponding to student ability and assessment properties such as difficulty. BKT does not capture assessment properties but employs a *dynamic* representation of student ability. A growing body of recent work has focused on modeling various structural properties of students and assessments in an attempt to combine the advantages of IRT and BKT, for instance [14, 15, 11, 5, 10, 12, 3]). In a recently proposed method known as Deep Knowledge Tracing (DKT) [16], a recurrent neural network was trained to predict student responses and was shown to outperform the best published results ([15]) on the publicly available ASSISTments data set [4] by about 20 percentage points with respect to the AUC metric described in Section 4.

To investigate DKT's advantage over traditional models, we compared a standard one parameter IRT model, two extensions of that model, and DKT on three data sets (two are publicly available and one is proprietary) on a realistic online prediction task that is typically required by computer-based learning systems (see Section 4), and which was consistent with the evaluation task employed in [16].[1] We reproduce the results of [16] on the ASSISTments data set, but find that proper accounting for duplicate data negates the claimed performance gains. For the two larger data sets, computational tractability hampered our ability to train DKT on fine-grained content labels, while training IRT-based models scaled to handle them. Moreover, the IRT-based models' best tractable performance matches or outperforms DKT's best tractable performance on all data sets, with a hierarchical extension of IRT performing the best in all cases. We conclude that for these data sets, IRT-based models provide simple, better-performing alternatives to DKT while also affording more interpretability and guarantees due to their formulation as Bayesian probabilistic models.

[*]Contributed equally to the work.

[†]Performed initial coding and analysis while at Knewton.

[1]Code for the IRT and DKT models, as well as instructions for reproducing our results, can be found at `github.com/Knewton/edm2016`.

2. MODELS OF STUDENT RESPONSES

In this section we set notation and describe the models we compare. Throughout, we will represent the student response data D as a set of tuples (s, i, r, t) indicating the student, item, correctness, and time of each response. In this paper, time will be indexed by interaction index (rather than wall clock time).

2.1 Item Response Theory (IRT)

Item Response Theory (IRT) is a standard framework for modeling student responses dating back to the 1950s [8, 13]. A single number, called the *proficiency* or *ability*, represents a student's knowledge state during the course of completing several assessments. It is assumed that this proficiency is not changing during this examination.[2]

The model assumes that many students have completed a test of dichotomous items and assigns each student s a proficiency $\theta_s \in \mathbb{R}$. A key innovation of IRT is to model variation across different items. In its simplest form, the *one-parameter model*, each item i is assigned a parameter β_i, representing the *difficulty* of the item. The probability that a student s answers item i correctly is given by $f(\theta_s - \beta_i)$, where f is some sigmoidal function.

When f is the logistic function, this corresponds to (structured) logistic regression, where the factors for a response to an item are indicators for students and items. We use a variant of this model known as 1PO (one-parameter ogive) IRT, where the link function $f(x) = \Phi(x)$ is the cumulative distribution function of the standard normal distribution[3]. The maximum likelihood solution of $\{\theta_s, \beta_i\}$ is underdetermined [4]; we take a Bayesian approach and regularize the solution of $\{\theta_s, \beta_i\}$ by imposing independent standard normal prior distributions over each θ_s and β_i.

2.1.1 Learning

To train the parameters on student response data, we maximize the log posterior probability of $\{\theta_s, \beta_i\}$ given the response data (the set of response correctnesses $\{r : (s, i, r, t) \in D\}$, each of which is 0 or 1). Assuming independent, standard normal priors on each θ_s, β_i, the log posterior is:

$$\log P(\{\theta_s\}, \{\beta_i\}|D) = \sum_{(s,i,r,t)\in D} r \log f(\theta_s - \beta_i) + (1-r)\log(1 - f(\theta_s - \beta_i)) - \frac{1}{2}\sum_s \theta_s^2 - \frac{1}{2}\sum_i \beta_i^2 + C\,. \quad (1)$$

We maximize this objective with respect to the parameters using standard second-order ascent methods to obtain the maximum a posteriori (MAP) estimate of each parameter.

2.2 Hierarchical IRT (HIRT)

In many situations, including each of our data sets, the assessment items may have structure that can inform predictions of student responses. For example, groups of items may assess the same topic, resulting in item properties that are more similar within groups than across them Alternatively, items may be derived from common templates. Templates, often found in math courses, look like "What is $x + y$?" and a particular instantiation is generated by choosing values for x and y. For example, the ASSISTments data set contains several *problems*, many of which are with the same *template*, many of which in turn assess a single *skill*.

We can augment the IRT model to incorporate knowledge about item groups, resulting in a hierarchical IRT model (HIRT). Each item i is associated with a group $j(i)$ whose difficulty is distributed normally around a per-group mean $\mu_{j(i)}$: $\beta_i \sim N(\mu_{j(i)}, \sigma^2)$. Each μ_j is in turn distributed according to the hyperprior $\mu_j \sim N(0, \tau^2)$. This reflects the belief that the difficulty of items in the same group should be similar. The degenerate cases provide some intuition: the limit $\sigma \to 0$ is the same model as 1PO IRT where we consider the items in the group to be the same item, and the limit $\tau \to 0$ is equivalent to a 1PO IRT model with no groupings.

2.2.1 Learning

Learning is done similarly to Bayesian IRT (section 2.1), except that we ascend the *modified* log posterior probability

$$\log P(\{\theta_s\}, \{\beta_i\}, \{\mu_t\}|D) = \sum_{(s,i,r,t)\in D} r \log f(\theta_s - \beta_i) + (1-r)\log(1 - f(\theta_s - \beta_i)) - \frac{1}{2}\sum_s \theta_s^2 - \frac{1}{2\sigma^2}\sum_i (\beta_i - \mu_{j(i)})^2 - \frac{1}{2\tau^2}\sum_j \mu_j^2 + C\,. \quad (2)$$

We maximize this objective with respect to $\{\theta_s, \beta_i, \mu_j\}$.

2.3 Temporal IRT (TIRT)

1PO IRT and HIRT assume each student's knowledge state remains constant over time. However, in a setting where a student may be acquiring (or forgetting) knowledge over a period of time (e.g., while interacting with a tutoring system), we can extend this model by modeling each θ_s as a stochastic process varying over time (see for example [5]). We adopt the approach described in [3], modeling the student's knowledge as a Wiener process:

$$P(\theta_{s,t+\tau}|\theta_{s,t}) = e^{-\frac{(\theta_{s,t+\tau} - \theta_{s,t})^2}{2\gamma^2\tau}} \;\forall s, t, \tau\,. \quad (3)$$

In other words, the change in student s's knowledge state between time t and a future time $t + \tau$ (expressed as $\theta_{s,t} - \theta_{s,t+\tau}$) is normally distributed about 0 with variance $\gamma^2\tau$ where γ is a parameter controlling the "smoothness" with which the knowledge state varies over time.

2.3.1 Learning

We fit the parameters according to the procedure described in [3]. Estimating the entire trajectory $\vec{\theta}_{s,t}$ for each student simultaneously with item parameters is very expensive and

[2] For an in depth discussion of IRT and a review of related literature see [17], especially Chapter 5.

[3] The ogive yields nearly identical results to the commonly used logistic link function, but allows closed-form posterior computation in the temporal IRT model described in Sec. 2.3

[4] For example, the response predictions are invariant when adding a constant offset to the $\{\theta_s\}$'s and $\{\beta_i\}$'s.

difficult to do in real-time. To simplify the approach, we learn parameters in two stages:

1. We learn the β_i according to a standard 1PO IRT model (see Section 2.1.1) on the training student population and freeze these during validation.
2. For each response of each student in the held-out validation population, we predict this response according to a temporal IRT model given the student's previous responses, as described below. For further details of the validation procedure, see Section 4.

For the second step, we combine the approximation:

$$P(\{(s', i, r, t') \in D : s' = s, t' \leq t\}|\theta_{s,t}) \approx \prod_{(s',i,r,t') \in D : s'=s, t' \leq t} P((s', i, r, t')|\theta_{s,t}) \quad (4)$$

with (3), integrating out previous proficiencies of the student to get a tractable approximation of the log posterior over the student's current proficiency given previous responses:

$$\log P(\theta_{s,t}|D) \approx \sum_{\substack{(s',i,r,t') \in D \\ s'=s, t' \leq t}} [r \log f(\tilde{\alpha}_{t'}(\theta_{s,t} - \beta_i)) + (1-r)\log(1 - f(\tilde{\alpha}_{t'}(\theta_{s,t} - \beta_i)))], \quad (5)$$

where $\tilde{\alpha}_{t'} = \left(1 + \gamma^2(t - t')\right)^{-1/2}$. The $\tilde{\alpha}_t$'s are essentially discounting the relative effect of older responses when estimating the current proficiency. See [3] for details.

2.4 Deep Knowledge Tracing (DKT)

Recently, a recurrent neural network was used to predict student responses [16]. Such architectures have seen enormous success in applications to a wide range of other domains (e.g., image processing [6], speech recognition [7], and natural language processing [20]).

In this model, the input vectors are representations of whether the student answered a particular question correctly or incorrectly at the previous time step, and the output vectors are representations of the probability, over all the questions in the question bank, that a student will get the question correct at the following time step. In [16], the authors propose using a one-hot vector $\vec{x}_{s,t} \in \mathbb{R}^{2I}$ to represent the response of a student s (on item i) at time t. Here I is the total number of items and the first I slots represent answering correctly and the remaining I slots represent answering incorrectly. Output vectors $\vec{y}_{s,t} \in \mathbb{R}^I$ are vectors of probabilities, where the ith element of $\vec{y}_{s,t}$ is the model's predicted probability that student s would answer item i correctly at time $t+1$.

We use a model with one hidden layer, of dimension H, which is fully connected[5] to both the input and output layers, as well as recurrently to itself. This model is able to capture temporal effects (via the recurrent component of the network) and remains flexible enough to describe non-trivial relationships between items.

[5]Note that in [16], an LSTM network was used in addition to the RNN described here, and the performance of the two networks was comparable.

2.4.1 Learning and Parameter Choices

In order to make learning tractable, we reduced the dimensionality of the input by projecting the $\vec{x}_{s,t} \in \mathbb{R}^{2I}$ to a lower dimensional space $\mathbb{R}^C$ using a random projection matrix $c : \mathbb{R}^{2I} \to \mathbb{R}^C$, as was done in [16]. We used batch gradient ascent with dropout [18], and chose the input dimensionality C and the hidden dimensionality H by sweeping these parameters on a data set that was held out from the data used for training and cross-validation.

The predictions are given by the following equations:

$$\vec{h}_{s,t+1} = g(W_{hh}\vec{h}_{s,t} + W_{xh}c(\vec{x}_{s,t}) + \vec{b}_h) \quad (6)$$

$$\vec{y}_{s,t+1} = \phi(W_{hy}\vec{h}_{s,t+1} + \vec{b}_y) \quad (7)$$

Here, g and ϕ are the logistic and arctangent functions, respectively. The parameters of the model $W_{hh}, W_{xh}, W_{hy}, \vec{b}_h, \vec{b}_y$ are fit by optimizing the cross-entropy of the responses with the predicted probabilities (which is equivalent to the log likelihood if these probabilities were produced via a generative probabilistic model):

$$\sum_{(s,i,r,t) \in D} r \log y_{s,t,i} + (1-r)\log(1 - y_{s,t,i}) \quad (8)$$

Stochastic gradient ascent with minibatches of students on the unrolled RNN, coded using Theano [1], was used to optimize this objective function.

3. DATA SETS

In order to test these models, we used three data sets, two publicly accessible and one proprietary. Each of these data sets comes from a system in which students interact with a computer-based learning system in a variety of educational settings (e.g., interspersed with classroom lectures, offline work, etc.).

3.1 ASSISTments

This data set comes from the ASSISTments product, an online platform which engages students with formative assessments replete with scaffolded hints. Most assessments are templated, and each problem is aligned with one, several, or none of the skills that the product is attempting to teach.

The data set [4] is divided in two parts, the "skill builder" set associated with formative assessment and the "non skill builder" set associated with summative assessment. All of our results are reported on the "skill builder" data set as we expect a stronger temporal signal from formative assessment than from summative assessment. This was also the evaluation data set for [16].

In preprocessing the data, we associated items not aligned with a skill to a designated "dummy" skill, as was done in [16]. We chose to discard rows duplicating a single interaction (represented by a unique `order_id` value), a step we do not believe was taken by [16]. These duplicate rows arise when a single interaction is aligned with multiple skills. Without removing these duplicates, models that process all skills simultaneously, including DKT and the IRT variants used in this paper, will see the same student interaction several times in a row, essentially providing these models

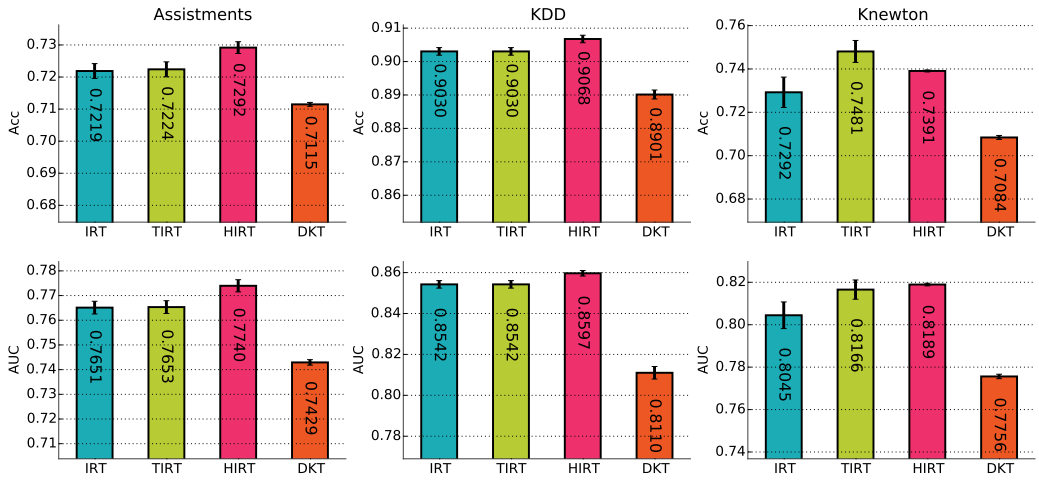


Figure 1: Summary of results across models and metrics. Error bars represent the standard error of measure of the metric across five folds. For TIRT, parameter selection yielded $\gamma^2 = 0.01$ for ASSISTments, $\gamma^2 = 0$ for KDD (making it identical to IRT), and $\gamma^2 = 100.0$ for Knewton. For HIRT, parameter selection yielded $\sigma^2 = 0.125$ and $\tau^2 = 0.5$ for ASSISTments, $\sigma^2 = 0.5$ and $\tau^2 = 0.25$ for KDD, and $\sigma^2 = 0.25$ and $\tau^2 = 0.125$ for Knewton. For DKT, $C = 50$, $H = 100$, and the probability of dropout is 0.25 for all models.

access to the ground truth when making their predictions. This can artificially boost prediction results by a significant amount (see Section 5), as these "duplicate" rows account for approximately 25% of the rows. Indeed, we observed that the performance gains of DKT are negated when these duplicates are removed (see Section 5). Note that typical BKT-based approaches are not susceptible to this artificial boost, since they usually split the data by skill and train separate models.

After pre-processing, the data set consisted of 346,740 interactions for 4,097 users on 26,684 items arising from 815 templates and 112 skills. The overall percent correct was 64.54%.

3.2 KDD Cup

In 2010, the PSLC DataShop released several data sets derived from Carnegie Learning's Cognitive Tutor in (Pre-)Algebra from the years 2005–2009 [19]. We used the largest of the "Development" data sets, labeled "Bridge to Algebra 2006–2007."

One distinct difference between Carnegie Learning's product and ASSISTments is that Carnegie Learning provides much finer representations of the concepts assessed by an individual item. In particular, Carnegie Learning is built around scaffolded, formative assessment, where each *step* a student takes to answer a *problem* is counted as a separate interaction, with each step potentially assessing different skills (called Knowledge Components (KCs) in the data set). Note that this "Problem → Step" structure provides a hierarchy which HIRT (Section 2.2) can exploit.

Like ASSISTments, any particular interaction may assess zero or more skills. We follow the same methodology as we did in Section 3.1, arbitrarily but consistently retaining only one of the skills after preprocessing, and associating items not associated with any skills with a designated "dummy" skill.

After pre-processing, the data set retained 3,679,198 interactions for 1,146 users on 207,856 steps arising from 19,355 problems and 494 KCs. The overall percent correct was 88.82%.

3.3 Knewton

Data was collected from a variety of educational products integrated with Knewton's adaptive learning platform and used in various classroom settings across the world. These products vary with respect to the educational content used (disciplines spanned math, science, and English language learning) as well as the way in which students are guided through the content. For example, students may take an initial assessment and then be remediated on areas needing improvement. In other products, students start from the beginning and work toward a predefined goal set by the teacher. In all of these settings, Knewton receives data about each interaction (the (s, i, r, t) tuple of Section 2). We utilized approximately 1M responses of 6.3K randomly sampled students on 105.6K questions spanning roughly 4 months. Students who worked on fewer than 5 questions total were excluded. After pre-processing, student history lengths ranged from 5 to 3.2K responses. The overall percent correct of these responses is 54.6%.

4. EVALUATION METHODOLOGY

4.1 Parameter Selection

For each data set, 20% of students were first set aside for parameter selection, which we performed as follows:

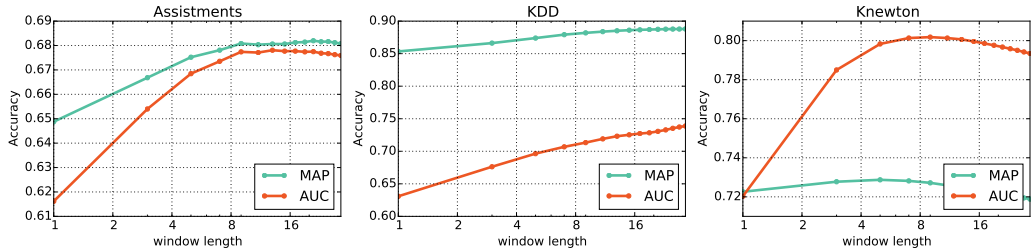


Figure 2: Accuracy metrics for the three data sets computed using a rolling window of previous responses, as a function of window length. Response accuracy is computed by predicting correct if the majority of responses in the window are correct.

	IRT	HIRT	tIRT	DKT*
ASSISTments	`problem_id`	`template_id` → `problem_id`	`problem_id`	`template_id`
KDD	`Step Name`	`Problem Name` → `Step Name`	`Step Name`	`KC`
Knewton	`item_id`	`concept_id` → `item_id`	`item_id`	`concept_id`

Table 1: Item labels yielding best results for each model and data set. For HIRT, the first label specifies the difficulty mean grouping identifier, and the second the item identifier.

- For 1PO IRT there were no parameters to select.
- For HIRT, we swept values of the variances τ^2 and σ^2 of the group means and item difficulties respectively, including regimes (τ^2 small) which made the model mathematically equivalent to 1PO IRT.
- For TIRT, we swept the temporal smoothness parameter γ^2, including the regime (γ^2 small) which made the model mathematically equivalent to 1PO IRT.
- For DKT, we swept the compression dimension C (the dimension of the space to which the input was projected using a random matrix), the hidden dimension H, the dropout probability p, and the step size of our gradient ascent.

4.2 Online prediction accuracy

We use an evaluation method we call *online response prediction* which matches that of [16]. Students are first split into training and testing populations. Each model is first trained on the training population and the model parameters that are not student-level (item parameters for IRT-based models, weights for neural networks) are frozen. Then for each time $t > 1$ in each testing student's history, we train the student-level parameters in the model on the first $t-1$ interactions of the student history and allow it to compute the probability that the t'th response is correct. This process mirrors the practical task that must be completed by an ITS.

We report two different metrics for comparing the predicted correctness probabilities with the observed correctness values. Accuracy (Acc) is computed as the percent of responses in which the correctness coincides with the probability being greater than 50%. AUC is the Area Under the ROC Curve of the probability of correctness for each response.

We use five-fold cross validation (by partitioning the students) on the 80% of the data set remaining after parameter selection (Section 4.1), averaging the Acc and AUC metrics over five different splits of the student population.

5. RESULTS AND DISCUSSION

Table 1 enumerates the fields chosen in each data set to identify items and item groups (for HIRT only) that yielded the computationally tractable model with the best results. Note that for the IRT-based models, our validation scheme (Section 4.2) estimates a single number θ_{st} for each student at each point $t > 1$ of the validation.For computational reasons, it was not feasible to evaluate DKT on fine-grained labels in KDD and Knewton (for ASSISTments, fine-grained labels were tractable but yielded worse results), whereas all IRT variants were able to process data at the finest levels.

We trained and validated each of the three models on each of the three data sets as described in Sec. 4. The results on our evaluation task are summarized in Figure 1. The results clearly indicate that simple IRT-based models do as well or significantly better than DKT across all data sets.

The fact that HIRT is the best-performing model across the board (except for MAP accuracy on the Knewton dataset where TIRT slightly outperforms it) suggests that grouping structure is useful information to exploit when predicting student responses. Indeed, the HIRT model does have access to strictly more information than the other models in that it has both the item and group identifier associated with each interaction. While the DKT model does have the ability to infer item relationships from data, our results indicate that building in this knowledge is more advantageous in a variety of educational settings. One potential area to explore is in learning a hierarchical model purely from the data, which could profit from the structured Bayesian framework without requiring prior information or expert labels.

The temporal IRT model yielded higher accuracy on the Knewton dataset, but not on the other two data sets. To understand these effects, we investigated the degree to which temporal structure in the data affects predictive performance

by looking at how a naive "windowed percent correct" (predict the student will answer the tth question correctly if they answer at least half of the previous w questions correctly) model performs as a function of window length w (Figure 2). The Knewton data set has a clear optimal window length – integrating over windows too short or too long degraded performance, which is indicative of nontrivial temporal structure. However, for the ASSISTments and KDD data sets, longer window lengths perform equal or better than shorter window lengths, suggesting that static models would do just as well in these cases. Indeed, this would explain why TIRT does more or less the same as baseline 1PO IRT on ASSISTments and KDD but shows significant improvement on the Knewton data set. However, it does not explain why DKT lags regardless of the amount of temporal structure.

Finally, we note that our DKT results in Figure 1 contradict those of [16] on the ASSISTments data set, which reported an AUC of 0.86. We believe this is due to data cleaning issues, specifically the issue of removing duplicates so as not to artificially boost online prediction accuracy, as discussed in Section 3.1. Indeed, we were able to reproduce the performance reported in [16] when applying our RNN implementation on the raw data set (with duplicates left in).

Other recent work [9] points out that the specific method of computing AUC in [16] also significantly affects the reported performance relative to BKT-based models, and further demonstrates that BKT-based models can perform just as well as DKT on a variety of data sets.

6. CONCLUSION

Our results indicate that simple IRT-based models equal or outperform DKT on a variety of data sets, suggesting that incorporating domain knowledge into structured Bayesian models comprises a promising area of future research for modeling student interaction data.

In our experience, structured models were easier to train and required less parameter tuning than DKT. Moreover, the computational demands of DKT hampered our ability to fully explore the parameter space, and we found that computation time and memory load were prohibitive when training on tens of thousands of items. These issues could not be mitigated by reducing dimensionality without significantly impairing performance. Further work on discriminative models is necessary to bridge this gap, but currently, IRT-based models seem superior both in terms of performance and ease of use, making them suitable candidates for real-world applications (e.g. intelligent tutoring systems, recommendation systems, or student analytics).

A promising avenue of research could explore combining the advantages of structured Bayesian models with those of large-scale discriminative models, which have provided superior performance in several other domains, particularly in the large-data regime. A crucial challenge for structured models is how to accommodate the diversity of educational settings from which the data are collected (different content, different classroom environments, etc.) while retaining the structure that drives predictive power and interpretability.

7. REFERENCES

[1] Bergstra, J., et al. Theano: a CPU and GPU math expression compiler. In *SciPy 2010*.

[2] Corbett, A., and Anderson, J. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction 4*, 4 (1995), 253–278.

[3] Ekanadham, C., and Karklin, Y. T-SKIRT: Online estimation of student proficiency in an adaptive learning system. *Machine Learning for Education Workshop at ICML* (2015).

[4] Feng, M., Heffernan, N., and Koedinger, K. Addressing the assessment challenge with an online system that tutors as it assesses. In *User Modeling, Adaption, and Personalization*, G.-J. Houben, G. McCalla, F. Pianesi, and M. Zancanaro, Eds. 2010, pp. 243–266.

[5] Gonzalez-Brenes, J., Huang, Y., and Brusilovsky, P. General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. In *EDM 2014*.

[6] Gregor, K., et al. DRAW: A recurrent neural network for image generation. In *ICML 2015*.

[7] Hinton, G., et al. Deep neural networks for acoustic modeling in speech recognition.

[8] Hulin, C. L., and Drasgow, F. Item Response Theory. In *Handbook of Industrial and Organizational Psychology*, S. Zedeck, Ed., vol. 1. American Psychological Association, 1990, pp. 577–636.

[9] Khajah, M., Lindsey, R. V., and Mozer, M. C. How deep is knowledge tracing? In *EDM 2016*.

[10] Khajah, M. M., Huang, Y., González-Brenes, J. P., Mozer, M. C., and Brusilovsky, P. Integrating knowledge tracing and item response theory. *Personalization Approaches in Learning Environments* (2014), 7.

[11] Lan, A. S., Studer, C., and Baraniuk, R. G. Time-varying learning and content analytics via sparse factor analysis. In *KDD 2014*.

[12] Lee, J. I., and Brunskill, E. The Impact on Individualizing Student Models on Necessary Practice Opportunities.

[13] Lord, F. M. *A Theory of Test Scores*. No. 7 in Psychometric Monograph. Psychometric Corporation, 1952.

[14] Pardos, Z. A., and Heffernan, N. T. Modeling individualization in a Bayesian Networks implementation of knowledge tracing. In *User Modeling, Adaption, and Personalization*, P. D. Bra, A. Kobsa, and D. Chin, Eds. 2010, pp. 255–266.

[15] Pardos, Z. A., and Heffernan, N. T. KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. In *User Modeling, Adaption, and Personalization*, J. A. Konstan, R. Conejo, J. L. Marzo, and N. Oliver, Eds. 2011, pp. 243–254.

[16] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., and Sohl-Dickstein, J. Deep Knowledge Tracing. In *NIPS 2015*.

[17] Rupp, A. A., Templin, J., and Henson, R. A. *Diagnostic Measurement: Theory, Methods, and Applications*. Guilford Press, 2010.

[18] Srivastava, N., et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research 15* (2014), 1929–1958.

[19] Stamper, J., Niculescu-Mizil, A., Ritter, S., G.J Gordon, G., and Koedinger, K. Challenge data sets from KDD Cup 2010. `pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp`.

[20] Vinyals, O., et al. Grammar as a foreign language. In *NIPS 2015*.

[21] Wilson, K. H., and Nichols, Z. The Knewton Platform: A General-Purpose Adaptive Learning Infrastructure.

Going Deeper with Deep Knowledge Tracing

Xiaolu Xiong, Siyuan Zhao, Eric G.
Van Inwegen, Joseph E. Beck
Worcester Polytechnic Institute
100 Institute Rd
Worcester, MA 01609
508-831-5000
{xxiong, szhao, egvaninwegen,
josephbeck}@wpi.edu

ABSTRACT

Over the last couple of decades, there have been a large variety of approaches towards modeling student knowledge within intelligent tutoring systems. With the booming development of deep learning and large-scale artificial neural networks, there have been empirical successes in a number of machine learning and data mining applications, including student knowledge modeling. Deep Knowledge Tracing (DKT), a pioneer algorithm that utilizes recurrent neural networks to model student learning, reports substantial improvements in prediction performance. To help the EDM community better understand the promising techniques of deep learning, we examine DKT alongside two well-studied models for knowledge modeling, PFA and BKT. In addition to sharing a primer on the internal computational structures of DKT, we also report on potential issues that arise from data formatting. We take steps to reproduce the experiments of Deep Knowledge Tracing by implementing a DKT algorithm using Google's TensorFlow framework; we also reproduce similar results on new datasets. We determine that the DKT findings don't hold an overall edge when compared to the PFA model, when applied to properly prepared datasets that are limited to main (i.e. non-scaffolding) questions. More importantly, during the investigation of DKT, we not only discovered a data quality issue in a public available data set, but we also detected a vulnerability of DKT at how it handles multiple skill sequences.

Keywords

Knowledge tracing, deep learning, recurrent neural networks, student modeling, performance factors analysis, data quality

1. INTRODUCTION

Deep Learning (DL) is an emerging approach within the machine learning research community. A series of deep learning algorithms have been proposed in recent years to move machine learning systems toward the discovery of multiple levels of representation and they already had important empirical successes in a number of traditional AI applications such as computer vision and natural language processing [8]. Much more recently, Google's deep learning networks [7] beat a top human player at the game of Go. Most research in deep learning (e.g. Google's deep learning algorithm) has been focused on the studies of artificial neural networks.

Deep knowledge tracing (DKT), the recent adoption of recurrent neural nets (RNNs) in the area of educational data mining, achieved dramatic improvement over well-known Bayesian Knowledge Tracing models (BKT) and the results of it have been demonstrated to be able to discover the latent structure in skill concepts and can be used for curriculum optimization [1].

Driven by both noble goals (testing the reproducibility of scientific findings) and some selfish ones (how did they do so much better at predicting student performance?!), we set out to take the theories, algorithms, and code from the DKT paper and apply them ourselves to the same data and more data sets. As to the goal of reproducing the findings, we were motivated by studies discussing the importance of reproducibility [5]. In addition to applying DKT to the same data, we also tested the algorithm on a different ASSISTments dataset (which covers data in 2014-2015 school year), as well as the one of data sets from KDD Cup 2010. In our experiments with the original DKT algorithm, we uncovered three aspects of the ASSISTments 2009-2010 data set that, when accounted for, drastically reduce the effectiveness of the DKT algorithm. These can broadly be summarized as 1). an error in reporting the data (wherein rows of data were randomly duplicated). 2). an inconsistency of skill tagging, and 3). the use of information ignored by PFA and BKT. We will discuss these three inconsistencies and their impacts on the prediction accuracies in section 3.

2. DEEP KNOWLEDGE TRACING AND OTHER STUDENT MODELING TECHNIQUES

When describing neural networks, the use of 'deep' conventionally refers to the use of multiple processing layers; the 'Deep' in DKT refers to the recurrent structure of the network and the 'depth' of information over time. This family of neural nets represents latent knowledge state, along with its temporal dynamics, using large vectors of artificial neurons, and allows the latent variable representation of student knowledge to be learned from data rather than hard-coded.

Typical RNNs suffer from the now famous problems of vanishing and exploding gradients, which are inherent to deep networks. Figure 1 shows an unrolled RNN; there are loops at hidden layers, allowing information to be retained; this is the 'depth' of an RNN. When building a deep neural net, the standard activation functions , and cumulative backpropagation error signals either shrink rapidly or grow out of bounds. i.e., they either decay or grow exponentially ('vanish' or 'explode'). Long short-term memory (LSTM) model [14] is introduced to deal with the vanishing gradient problem; it also achieves remarkable results on many previously un-learnable tasks. LSTM, a variation of recurrent neural networks, contains LSTM units in addition to regular RNN units. LSTM units have two unique gates: forget and input gates

to determine when to forget previous information, and which current information is important to remember.

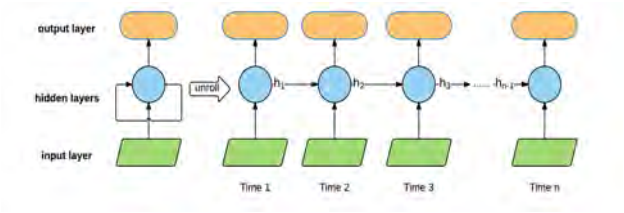


Figure 1. An unrolled Recurrent Neural Network (RNN)

The idea behind LSTM is simple. Some of the units are called constant error carousels (CEC). Each CEC uses an activation function *f*, the identity function, and has a connection to itself with fixed weight of 1.0. Due to *f*'s constant derivative of 1.0, errors backpropagated through a CEC cannot vanish or explode but stay the same magnitude. CECs are connected to several nonlinear adaptive units needed for learning nonlinear behavior. Weight changes of these units often profit from error signals, which propagate far back in time through CECs. CECs are the main reason why LSTM nets can learn to discover the importance of (memorize) events that happened thousands of discrete time steps ago while previous RNNs routinely fail in cases of minimal time lags of 10 steps. LSTM learns to solve many previously un-learnable DL tasks and clearly outperformed previous RNNs on tasks both in terms of reliability and speed [1].

In the DKT algorithm, at any time step, the input to RNNs is the student performance on a single problem of the skill that the student is currently working on. Since RNNs only accept fixed length of vectors as the input, we used one-hot encoding to convert student performance into fixed length of vectors whose all elements are 0s except for a single 1. The single 1 in the vector indicates two things: which skill was answered and if the skill was answered correctly. This data presentation draws a clear distinction between DKT and other student modeling methods, such as Bayesian Knowledge Tracing and Performance Factor Analysis.

The Bayesian Knowledge Tracing (BKT) model [10] is a 2-state dynamic Bayesian network where student performance is the observed variable and student knowledge is the latent data. The model takes student performances and uses them to estimate the student level of knowledge on a given skill. The standard BKT model is defined by four parameters: initial knowledge and learning rate (learning parameters) and slip and guess (mediating parameters). The two learning parameters can be considered as the likelihood the student knows the skill before he even starts on an assignment (initial knowledge, K_0) and the probability a student will acquire a skill as a result of an opportunity to practice it (learning rate). The guess parameter represents the fact that a student may sometimes generate a correct response in spite of not knowing the correct skill. The slip parameter acknowledges that even students who understand a skill can make an occasional mistake. Guess and slip can be considered analogous to false positive and false negative. BKT typically uses the Expectation Maximization algorithm to estimate these four parameters from training data. Based on the estimated knowledge, student performance at a particular practice opportunity can be calculated except the very first one, which only apples the value of K_0.

Skills vary in difficulties and amount of practices needed to master, so values for four BKT parameters are skill dependent. This lead to one major weakness of BKT [11]: it lacks the ability to handle multi-skill questions since it works by looking at the historical observation of a skill and cannot accommodate all skills simultaneously. One simple workaround is treating the multiple skill combination as a new joint skill and estimate a set of parameters for this new skill. Another common solution to this issue is to associate the performance on multiple skill questions with all required skills, by listing the performance sequence repeatedly [12]. This makes the model see this piece of evidence multiple times for each one of required skills. As a result, a multiple skill question is multiple single skill questions.

Another popular student modeling approach is the Performance Factors Analysis Model (PFA) [9]. PFA is a variant of learning decomposition, based on a reconfiguration of Learning Factor Analysis. Unlike, BKT, it has the ability to handle multiple skill questions. Briefly speaking, it uses the form of the standard logistic regression model with the student performance as the dependent variable. It reconfigures LFA (Learning factors analysis) [13] on its independent variables, by dropping the student variable and replaces the skill variable with question identity. This model estimates parameters for each item's difficulty and also two parameters for each skill reflecting the effects of the prior correct and incorrect responses achieved for that skill. Previous work that compares KT and PFA have shown that PFA to be the superior one [11]. One reason is the richer feature set that PFA can utilize and the fact that learning decomposition models are ensured to reach global maxima while the typical fitting approach of BKT is no guarantee of finding a global, rather than a local maximum.

Beside the theoretical comparison of DKT, BKT, and PFA, we can also compare them visually by looking at the differences between them in terms of inputs data. Consider a simple scenario that a student answers two questions from two skills each, Tables 1-3 compare different training data formats for these three modeling methods under that same scenario of student responses.

Table 1. An example of BKT's training data

Model ID	Skill ID	Response Sequence
1	A	1,0
2	B	0,1

Table 2. An example of PFA's training data

Index ID	Skill ID	Prior Correct	Prior Incorrect	Difficulty	Correct
1	A	0	0	0.7	1
2	A	1	0	0.75	0
3	B	0	0	0.6	0
4	B	0	1	0.65	1

Table 3. An example of DKT's training data

Index ID	One-hot encoding
1	1,0,0,0
2	0,0,1,0
3	0,0,0,1
4	0,1,0,0

3. METHODOLOGY AND DATA SETS

3.1 Implementation of Deep Knowledge Tracing in Tensorflow

The original version of DKT (Lua DKT[1]) was implemented in Lua scripting language using Torch framework and its source code has been released to the public. In order to have a comprehensive understanding of the DKT model, we decided to replicate and implement DKT model in Python and utilize Google's TensorFlow API [3] to help us with building neural networks. TensorFlow is Google Brain's second generation machine learning interface; it is flexible and can be used to express a wide variety of algorithms.

Our implementation of DKT in TensorFlow (TensorFlow DKT[2]) can be described as a directed graph, which is composed of a set of nodes. The graph represents a data flow computation, with extensions for allowing certain nodes to maintain and update persistent state and for branching and looking control, this is crucial for allowing RNN nodes to work on sequential data. In the directed graph, each node has zero or more inputs and zero or more outputs and represents the instantiation of an operation. An operation represents an abstract computation. In our implementation of DKT model, we adapted the loss function of the original DKT algorithm. It has 200 fully-connected hidden nodes in the hidden layer. To speed up the training process, we used mini-batch stochastic gradient descent to minimize the loss function. The batch size for our implementation is 100. For one batch, we randomly select data from 100 students in our training data. After the batch finishes training, 100 students in the batch are removed from the training data. We continue to train the model on next batch until all batches are done. Just as in the original Lua implementation, Dropout [4] was also applied to the hidden layer to avoid over-fitting.

4. DATA SETS

4.1 ASSISTments 2009-2010 Data Set

The original DKT paper conducted one of three of experiments using the ASSISTments 2009-2010 skill builder data set [16]. This data set was gathered from ASSISTments' skill builder problem sets, in which a student achieves mastery by working on similar (often isomorphic) questions until they can correctly answer n right in a row (where n is usually 3). After mastery, students do not commonly rework the same skill. This dataset contains 525,535 rows of student responses; there are 4,217 student ID's and 124 skills. Lua DKT achieved an AUC of 0.86 and noticeably outperformed BKT (AUC = 0.67) on this data set. However, during our investigation on the DKT source code and application, we believe we discovered three issues that have unintentionally inflated the performance of Lua DKT. These issues are:

4.1.1 Duplicated records

To our surprise and dismay, we found that the ASSISTments 2009-2010 data set has a serious issue of quality: large chunks of records are duplications that should not be there for any reason (e.g. see records of order id 36369610). These duplicated rows have the same information but only differ on the "opportunity" and "opportunity_original"; these two features record the number of opportunities a student has practiced on a skill and the number of practices on main problems of a skill respectively. It is impossible to have more than one 'opportunity' count for a single order id. This is definitely an error in the data set and these duplicated records should not be used in any analysis or modeling studies. We counted there are 123,778 rows of duplications out of 525,535 in the data set (23.6%). The existence of duplicated data is an avoidable oversight and ASSISTments team has acknowledged this error on their website. All new experiments in this work and following discussions exclude data of these duplications.

4.1.2 Mixing main problems with scaffolding problems

A mastery learning problem set normally contains over a hundred of main problems, and each main problem may have multiple associated scaffolding problems. Scaffolding problems were designed to help students acquire an integrated set of skills through processes of observations and guided practice; they are usually tagged with different skills and have different designs from the main problems. Because of the difference in usage, scaffolding questions should not be treated as the same as main problems. Student modeling methods such as BKT and PFA exclude scaffolding features. The experiment conducted by Lua DKT did not filter out scaffolding problems. This means that Lua DKT had the advantage of additional information; thus, the prediction results cannot be compared fairly with BKT. There are 73,466 rows of records of scaffolding problems.

4.1.3 Repeated response sequences with different skill tagging (Duplication by skill tag)

The 2009-2010 skill builder dataset was created as a subset of the 2009-2010 full dataset. The full dataset from 2009-2010 includes student work from both skill builder assignments (where a student works until a mastery threshold is reached) and more traditional assignments (where a student has a fixed number of problems). Any problem (or assignment) can be tagged with any number of skill tags. Typically, problems have just one skill tag; they seldom are tagged with two skills; they are very rarely tagged with three or more. Depending on the design of the content creator, a problem set may have multiple skill tags; many assignments - especially skill builders - will have the same skill tag for all problems. When the full dataset was decomposed into only mastery style assignments, the problems, and assignments that were tagged with multiple skills were included with a single tag, but repeated for each skill. This means that the sequence of action logs from one student working on one assignment was now repeated once per skill. For models such as RNNs that operate over sequences of vectors and memory on the entire history of

[1] https://github.com/chrispiech/DeepKnowledgeTracing

[2] https://github.com/siyuanzhao/2016-EDM

previous inputs, the issue of duplicated sequences is going to add additional weight onto the duplicated information; this will have undesired effects on RNN models.

For an example, suppose we have a hypothetical scenario that a student answers two problems which have been tagged with skill "A" and "B"; he answers first one correctly and the next one incorrectly. Table 4 shows the data set where responses have been repeated on skill "A" and "B". This format of data can be used in BKT models since BKT can build two models for skill "A" and "B" separately. When applying this sequential data set to DKT, we believe DKT can recognize the pattern when a problem tagged with skill "B" follows a problem tagged with "A"; the skill "B" problem has an extremely high chance to repeat skill "A" problem's response correctness. Note that skill ID can be mapped to skill names, but the order of skill ID is completely arbitrary.

Table 4. An example of repeated multiple-skill sequence

Index ID	Skill ID	Problem ID	Correctness
1	A	3	1
1	B	3	1
2	A	4	0
2	B	4	0

One approach to change the way of how multiple-skill problems are handled is to simply use the combination of skills as a new joint skill. Table 5 shows the data set which uses a joint skill of A and B. In this case, DKT no longer has access to repeated information. PFA and BKT can also adapt this format of data too.

Table 5. An example of joint skills on multiple-skill problems

Index ID	Skill ID	Problem ID	Correctness
1	A, B	3	1
2	A, B	4	0

Table 6. Three variants of ASSISTments 2009-2010 Data set

	09-10 (a)	09-10 (b)	09-10 (c)
Has duplicated records	No	No	No
Has scaffolding problems	Yes	No	No
Repeated multiple-skill sequences	Yes	Yes	No
Joint skills from multiple-skill	No	No	Yes

In order to understand the impact of having scaffolding problems and two approaches to dealing with multiple-skill problems, we generate three different data sets (namely 09-10 (a), 09-10 (b), 09-10 (c)) derivate from the ASSISTments 2009-1010 data set, as summarized in Table 6.

4.2 ASSISTments 2014-2015 Data Set

Even without the issue of duplicate rows, 2009-2010 skill builder set has lost its timeliness and certainly cannot represent the latest student data in an intelligent tutoring system. So we gathered another data set that covers 2014-2015 school years' student response records [16]. In this experiment, we randomly selected 100 skills from this year's data records. This data set contains 707,944 rows of records; each record represents a response to a main problem in a mastery learning problem set. Each problem set has only one associated skill and we take caution to make sure there is no duplicated row in this data set. We suspect this new data set contains different information that covers student learning patterns, item difficulties and skill dependencies.

4.3 KDD Cup 2010 Data Set

Our last data set comes from the Cognitive Algebra Tutor 2005-2006 Algebra system [6]. This data was provided as a development dataset in the KDD Cup 2010 competition. Although both ASSISTments and Cognitive Algebra Tutor involve using mathematics skills to solve problems, they are actually rather different from each other. ASSISTments serves primarily as computer-assisted practice for students' nightly homework and review lessons while the Cognitive Tutor is part of an integrated curriculum and has more support for learners during the problem-solving process. Another difference in terms of content structure is that the Cognitive Tutor presents a problem to a student that consists of questions (also called steps) of many skills. The Cognitive Tutor uses Knowledge Tracing to determine when a student has mastered a skill. A problem in the tutor can consist of questions of different skills, once a student has mastered a skill, as determined by KT, the student no longer needs to answer questions of that skill within a problem but must answer the other questions which are associated with the un-mastered skills. The number of skills in this dataset is substantially larger than the ASSISTments dataset [15]. One issue of using KDD data on PFA is how to estimate item difficulty feature. In this work, we use a concatenation of problem name and step name. However many such pairs are only attempted by 1 student and the difficulty values of these items are either 1.0 or 0.0, leading to both over-fitting and data leakage. To fix that, we replace difficulty values of these items with skills' difficulty information. Filtering out rows with missing values resulting in 607,026 rows of data with students responded correctly at 75.5% of the time. This KDD data set has 574 students worked on 436 skills in mathematics. The complete statistic information of five data sets can be found in Table 7.

Table 7. Data set statistics

	# records	# Students	# Skills
09-10 (a)	401,757	4,217	124
09-10 (b)	328,292	4,217	124
09-10 (c)	275,459	4,217	146
14-15	707,944	19,457	100
KDD	607,026	574	436

5. RESULTS

Student performance predictions made by each model are tabulated and the accuracy was evaluated in terms of Area Under Curve (AUC) and the square of Pearson correlation (r^2). AUC and r2 provide robust metrics for evaluation predictions where the value being predicted is either a 0 or 1 also represents different information on modeling performance. An AUC of 0.50 always represents the scored achievable by random chance. A higher AUC score represents higher accuracy. r^2 is the square of Pearson correlation coefficient between the observed and predicted values of dependent variable. In the case of r^2, it is normalized relative to the variance in the data set and it is not directly a measure of how good the modeled values are, but rather a way of measuring the proportion of variance we can explain using one or more variables. r^2 is similar to root mean squared error (RMSE) but is more interpretable. For example, it is unclear whether an RMSE of 0.3 is good or bad without knowing more about the data set. However, an r2 of 0.8 indicates the model accounts for most of the variability in the data set. Neither AUC nor r^2 method is a perfect evaluation metric, but, when combined, they account for different aspects of a model and provide us a basis for evaluating our models.

Experiments on every data set have been 5-fold student level cross-validated and all parameters are learned from training data. We used EM to train BKT and the limit of iteration was set to 200. Besides the number of hidden nodes and the size of mini-batch parameters we have discussed, we set the number of epochs of DKT to 100.

The cross-validated model predictions results are shown in Table 8 and Table 9. As can be seen, DKT clearly outperforms BKT on all data sets, but the results are no longer overwhelmingly in favor of DKT (both implementations). Note that Lua DKT implementation which we can access uses regular RNN nodes; TensorFlow DKT uses LSTM nodes.

Table 8. AUC results

	Torch DKT	TensorFlow DKT	PFA	BKT
09-10 (a)	0.79	0.81	0.70	0.60
09-10 (b)	0.79	0.82	0.73	0.63
09-10 (c)	0.73	0.75	0.73	0.63
14-15	0.70	0.70	0.69	0.64
KDD	0.79	0.79	0.71	0.62

Table 9. r^2 results

	Lua DKT	TensorFlow DKT	PFA	BKT
09-10 (a)	0.22	0.29	0.11	0.04
09-10 (b)	0.22	0.31	0.14	0.07
09-10 (c)	0.14	0.18	0.14	0.07
14-15	0.10	0.10	0.09	0.06
KDD	0.21	0.21	0.10	0.05

On the ASSISTments data sets, average DKT prediction performance across two implementations is better than PFA and it is not affected by removing scaffolding, as we change dataset from 09-10 (a) to 09-10 (b). On the other hand, PFA's performance increases from 0.70 to 073 in AUC and 0.11 to 0.14 in r^2 ($p \leq 0.05$), we believe that removing scaffolding helps reducing noise from data and provides PFA with a dataset with lower variance. When we switch to dataset 09-10 (c) where multiple skills were combined into joint skills, the performance of DKT suffers a noticeable hit, average AUC and average r^2 drop from 0.81 to 0.74 and from 0.30 to 0.18 respectively. This observation confirms our suspicion on repeated response sequence inflating the performance of DKT models. On the 09-10 (c) dataset and 14-15 dataset where no repeated response sequences and scaffolding problems, we notice that PFA performs as well as DKT.

A deeper way of looking at the impact of repeated response sequences on data set 09-10 (b) is splitting the prediction results into two, the predictions of leading records and repeated data points. We see that predictions on repeated data points (e.g. skill "B" problems in Table 4) have nearly perfect performance metrics (AUC = 0.97, r^2 = 0.74). On the other hand, the leading records (e.g. skill "A" problems in Table 4) have much lower prediction results (AUC = 0.77, r^2 = 0.23). That said, we also notice these numbers are still higher than 09-10 (c)'s results, which uses joint skill tags to avoid repeated sequences. One can explain this as making DKT to model skills individually can cause data duplications but it also can have benefits on building skill dependencies over time and use such information to make better predictions.

On the KDD dataset, the performance results of two DKT implementations are definitely better than both BKT and PFA ($p \leq 0.05$). There are a few possible reasons for this performance gap between PFA and DKT. First of all, as we have mentioned, we have to adjust item difficulty values for many problems in order to avoid overfitting and data leakage, which leads to the lower predictive power of that feature and lower PFA performance. Another possible explanation of DKT is winning on KDD data set is that DKT can better exploit step responses. The structure of KDD data set made it is difficult to distinguish "main problems" and "scaffolding problems", thus PFA is unable to have a more unified data set for this part of the experiment. That said, the advantage of DKT shows its power on complicated and realistic data sets.

6. DISCUSSION AND CONTRIBUTION

Within this paper, we have compared two well-studied knowledge modeling methods with the emerging Deep Knowledge Tracing algorithm. We have compared these models in terms of their power of predicting student performance in 5 different data sets. Contrary to our expectation, the DKT algorithm did not achieve overwhelmingly better performance when compared to PFA model on ASSISTments data sets when they are properly prepared. DKT appears to perform much better on KDD dataset, but we believe this is due to PFA model undermined by inaccurate item difficulty estimation.

A second interesting finding is that when DKT is fed repeated response sequences derived from the transformation of problems tagged with multiple skills, the overall performance of DKT is certainly better than PFA and BKT. Our explanation is that DKT's implementation backbone, RNNs, has the power of

remembering exact patterns of sequential data and could thus inflate prediction performance on responses tagged with multiple skills and repeated per skill. More discussion and special attention are required when handling multiple skill problems in DKT algorithm.

Last, but not least, during the investigation of DKT, we discovered an issue in data quality arising from duplicated information in a publicly available data set. The duplication issues (caused by unclear transformational rules and some other as-of-yet-to-be-ascertained cause) allowed us a natural experiment to examine the impact of duplications on the robustness of these algorithms. These discoveries (the data duplications and their subsequent impact) should serve as a reminder of the importance of data preprocessing and transformation procedures in the work of knowledge discovery and data mining. Or, put another way, while we advance new algorithms and fine tune their parameters, we should also consider (and, if possible, report on) the robustness of the algorithms to common data glitches.

7. FUTURE WORK AND CONCLUSION

There are several directions for further research in the area of DKT modeling. Prior work [2] has shown that the use of context-dependent RNN language model improved the performance in the task of the Wall Street Journal speech recognition task. More features like student features (e.g. prior knowledge, completion rates, time on learning, etc.), and content features (problem difficulty, skill hierarchies, etc.) may be available and could be used. A context-dependent DKT implementation could be created by adding an extra input vector containing these features.

Another open area for future work is that DKT and other deep learning algorithms are not restricted to one kind of output or application. It is also possible that we could apply deep learning algorithms on other modeling challenges such as wheel spinning, mastery speed, and affect detection.

In conclusion, our work here focuses on a primitive investigation of DKT and aims to provide us deeper insight on how DKT works. Overall, this paper suggests that DKT remains a promising approach to modeling student knowledge; however, we see that data which contains problems tagged with multiple skills has to be dealt carefully in DKT modeling. But, considering that this implementation of DKT: a) only relied on the sequences of student responses (just as BKT does) and no other information on skills and problems and b) performs substantially better than BKT and as good as PFA, we believe that DKT has great potential to outperform other methods when it utilizes more features.

8. ACKNOWLEDGEMENTS

We thank multiple current NSF grants (ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428 & DRL-1031398), the US Dept. of Ed (IES R305A120125 & R305C100024 and GAANN), and the ONR.

9. REFERENCES

[1] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., & Sohl-Dickstein, J. (2015). Deep Knowledge Tracing. In Advances in Neural Information Processing Systems (pp. 505-513).

[2] Mikolov, T., & Zweig, G. (2012, July). Context dependent recurrent neural network language model. In SLT (pp. 234-239).

[3] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., & Ghemawat, S. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. White paper, Google Research.

[4] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958.

[5] Open Science Collaboration. (2015). Estimating the reproducibility of psychological science. Science, 349(6251), aac4716.

[6] Stamper, J., Niculescu-Mizil, A., Ritter, S., Gordon, G.J., & Koedinger, K.R. (2010). Algebra I 2005-2006. Challenge data set from KDD Cup 2010 Educational Data Mining Challenge. Find it at http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp

[7] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., ... & Dieleman, S. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587), 484-489.

[8] Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009, June). Curriculum learning. In Proceedings of the 26th annual international conference on machine learning (pp. 41-48). ACM.

[9] Pavlik Jr, P. I., Cen, H., & Koedinger, K. R. (2009). Performance Factors Analysis-A New Alternative to Knowledge Tracing. Online Submission.

[10] Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction, 4(4), 253-278.

[11] Gong, Y., Beck, J. E., & Heffernan, N. T. (2010, June). Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In Intelligent tutoring systems (pp. 35-44). Springer Berlin Heidelberg.

[12] Heathcote, A., Brown, S., & Mewhort, D. J. K. (2000). The power law repealed: The case for an exponential law of practice. Psychonomic bulletin & review, 7(2), 185-207.

[13] Cen, H., Koedinger, K., & Junker, B. (2006, June). Learning factors analysis–a general method for cognitive model evaluation and improvement. In Intelligent tutoring systems (pp. 164-175). Springer Berlin Heidelberg.

[14] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[15] Pardos, Z. A., & Heffernan, N. T. (2011). KT-IDEM: Introducing item difficulty to the knowledge tracing model. In User Modeling, Adaption and Personalization (pp. 243-254). Springer Berlin Heidelberg.

[16] ASSISTments Data. (2015). Retrieved March 07, 2016, from https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data/skill-builder-data-2009-2010

Boosted Decision Tree for Q-matrix Refinement

Peng Xu
Polytechnique Montreal
peng.xu@polymtl.ca

Michel C. Desmarais
Polytechnique Montreal
michel.desmarais@polymtl.ca

ABSTRACT

In recent years, substantial improvements were obtained in the effectiveness of data driven algorithms to validate the mapping of items to skills, or the Q-matrix. In the current study we use ensemble algorithms on top of existing Q-matrix refinement algorithms to improve their performance. We combine the boosting technique with a decision tree. The results show that the improvements from both the decision tree and Adaboost combined are better than the decision tree alone and yield substantial gains over the best performance of individual Q-matrix refinement algorithm.

1. INTRODUCTION

A Q-matrix, as proposed by Tatsuoka (Tatsuoka, 1983), is a term commonly used in the literature of psychometrics and cognitive modeling that refers to a binary matrix which shows a correspondence between items and their latent attributes. Items can be questions or exercises proposed to students, and latent attributes are skills needed to succeed these items. Usually, a Q-matrix is defined by a domain expert. However, this task is non trivial and there might be errors, which in turn will result in erroneous diagnosis of students knowledge states (Rupp & Templin, 2008; Madison & Bradshaw, 2015). Therefore, better means to validate a Q-matrix is a highly desirable goal.

A fair number of algorithms have emerged in the last decade to validate an expert given Q-matrix based on empirical data (see for eg. recent work from Chen, Liu, Xu, & Ying, 2015; de la Torre & Chiu, 2015; Durand, Belacel, & Goutte, 2015). Desmarais, Xu, and Beheshti (2015) showed that Q-matrix refinement algorithms can be combined using an ensemble learning technique. They used a decision tree and the results show a substantial and systematic performance gain over the best algorithm, in the range of 50% error reduction for real data, even though the best algorithm is not always the same for different Q-matrices.

The encouraging the results obtained by combining the output of Q-matrix refinement algorithms leads us to pursue further along the line of using ensemble learning, or meta-learning techniques. In particular, a common approach is to use boosting with a decision tree algorithm. This is the approach explored in the current study.

2. THREE TECHNIQUES TO Q-MATRIX VALIDATION

Our approach relies on meta-learning algorithms whose principle in a general way is to combine the output of existing algorithms to improve upon the individual or average results. In our case, the approach combines a decision tree trained on the output of Q-matrix validation algorithms with boosting, a weighted sampling process in the training of the decision tree to improve its accuracy. In this section, we first describe the Q-matrix validation techniques before describing the decision tree and boosting algorithms.

2.1 minRSS

The first Q-matrix refinement technique that serves as input to the decision tree is from Chiu and Douglas (2013). We name this technique minRSS. The underlying cognitive model behind minRSS is the DINA model(see De La Torre, 2009).

For a given Q-matrix, there is a unique and ideal response pattern for a given a student skills mastery profile. That is, if there are no slip and guess factors, then the response pattern for every category of student profile is fixed. The difference between the real response pattern and the ideal response pattern represents a measure of fit for the Q-matrix, typically the Hamming distance. Chiu and Douglas (2013) considered a more refined metric. The idea is if an item has a smaller variance (or entropy), then it should be given a higher weight in measure of fit. A first step is to compute the ideal response matrix for all possible student profile, and then to find the corresponding student profiles matrix A given observed data. First, a squared sum of errors for each item k can be computed by

$$RSS_k = \sum_{i=1}^{N} (r_{ik} - \eta_{ik})^2$$

where r is the real response vector while η is the ideal response vector, and N is the number of respondents. Then, the worst fitted item (highest RSS) is chosen to update its correspondent q-vector. Given all permutations of the skills for a q-vector, the q-vector with the lowest RSS is chosen to

replace the original one. The Q-matrix is changed and the whole process repeated, but the previously changed q-vector is eliminated from the next iteration. The whole procedure terminates when the RSS for each item no longer changes. This method was shown by Wang and Douglas (2015) to yield good performance under different underlying conjunctive models.

2.2 maxDiff

Akin to minRSS, the maxDiff algorithm relies on the DINA model. De La Torre (2008) proposed that a correctly specified q-vector for item j should maximize the difference of probabilities of correct response between examinees who have all the required attributes and those who do not. A natural idea is to test all q-vectors to find that maximum, but that is computationally expensive. De La Torre (2008) proposed a greedy algorithm that adds skills into a q-vector sequentially. Assuming δ_{jl} represents the difference to maximize, the first step is to calculate δ_{jl} for all q-vectors which contains only one skill and the one with biggest δ_{jl} is chosen. Then, δ_{jl} is calculated for all q-vectors which contains two skills including the previously chosen one. Again the q-vector with the biggest δ_{jl} is chosen. This whole process is repeated until no addition of skills increases δ_{jl}. However, this algorithm requires knowing slip and guess parameters of the DINA model in advance. For real data, they are calculated by EM (Expectation Maximization) algorithm (De La Torre, 2009).

2.3 ALSC

ALSC (Conjunctive Alternating Least Square Factorization) is a common matrix Factorization (MF) algorithm. Desmarais and Naceur (2013) proposed to factorize student test results into a Q-matrix and a skills-student matrix with ALSC.

ALSC decomposes the results matrix $\mathbf{R}_{m\times n}$ of m items by n students as the inner product two smaller matrices:

$$\neg\mathbf{R} = \mathbf{Q}\,\neg\mathbf{S} \tag{1}$$

where $\neg\mathbf{R}$ is the negation of the results matrix (m items by n students), $\mathbf{Q}$ is the m items by k skills Q-matrix, and $\neg\mathbf{S}$ is negation of the the mastery matrix of k skills by n students (normalized for rows columns to sum to 1). By negation, we mean the 0-values are transformed to 1, and non-0-values to 0. Negation is necessary for a conjunctive Q-matrix. As such, the model of equation (1) is analogous to the DINA model without a slip and guess parameter.

The factorization consists of alternating between estimates of $\mathbf{S}$ and $\mathbf{Q}$ until convergence. Starting with the initial expert defined Q-matrix, $\mathbf{Q}_0$, a least-squares estimate of $\mathbf{S}$ is obtained:

$$\neg\hat{\mathbf{S}}_0 = \left(\mathbf{Q}_0^{\mathrm{T}}\,\mathbf{Q}_0\right)^{-1}\mathbf{Q}_0^{\mathrm{T}}\,\neg\mathbf{R} \tag{2}$$

Then, a new estimate of the Q-matrix, $\hat{\mathbf{Q}}_1$, is again obtained by the least-squares estimate:

$$\hat{\mathbf{Q}}_1 = \neg\mathbf{R}\,\neg\hat{\mathbf{S}}_0^{\mathrm{T}}\left(\neg\hat{\mathbf{S}}_0\,\neg\hat{\mathbf{S}}_0^{\mathrm{T}}\right)^{-1} \tag{3}$$

And so on until convergence. Alternating between equations (2) and (3) yields progressive refinements of the matrices $\hat{\mathbf{Q}}_i$ and $\hat{\mathbf{S}}_i$ that more closely approximate $\mathbf{R}$ in equation (1). The final $\hat{\mathbf{Q}}_\mathbf{i}$ is rounded to yield a binary matrix.

3. DECISION TREE

The three algorithms for Q-matrix refinement described in the last section are to be combined to yield with a decision tree to obtain an improved refinement recommendation, and further improved by boosting. We describe the decision tree before moving on to the boosting method.

Decision tree is a well-know technique in machine learning and it often serves as an ensemble learning algorithm to combine individual models into a more powerful model. It uses a set of feature variables (individual model predictions) to predict a single target variable (output variable). There are several decision tree algorithms, such as ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993), CART (Breiman, Friedman, Stone, & Olshen, 1984). We used `rpart` function from the R package of the same name (Therneau, Atkinson, & Ripley, 2015). It implements the CART algorithm. This algorithm divides the learning process into two phases. The first phase is for feature selection, or tree growing, during which the feature variables are chosen sequentially according to *Gini impurity* (Murphy, 2012). Then in the second phase, the pruning phase, deep branches are split into wider ones to avoid overfitting.

A decision tree is a supervised learning technique and therefore requires training data. To obtain training data of sufficient size, Desmarais et al. (2015) use synthetic data from Q-matrices generated by random permutations of the perturbated Q-matrix. Since the ground-truth Q-matrix of synthetic data is known, it becomes possible to generate training data containing the class label. The training set for decision tree can take this form:

	Algorithm target prediction			Other factors
Target	minRSS	maxDiff	ALSC	...
1	1	0	1	...
0	0	1	0	...
...	...	...	...	...

The other factors considered to help the decision tree to improve prediction are the number of skills per row (SR), number of skills per column (SC). Moreover, a feature named *stickiness* is introduced and makes a critical difference. It measures the rigidity of cells under each validation methods. Stickiness represents the rate of a given algorithm's false positives for a given cell of a Q-matrix. The rate is measured by "perturbating" in turn each and every cell of the Q-matrix, and by counting the number of times the cell is a false positive. The decision tree can use the stickiness factor as an indicator of the reliability of a given Q-matrix refinement algorithm suggested value for a cell. Obviously, if a cell's stickiness value is high, the reliability of the corresponding algorithm's refinement will be lower.

4. BOOSTING

The current work extends the idea of using a decision tree with another meta-learning technique named boosting.

Boosting (Schapire & Freund, 2012) serves as a meta-learning technique for lifting a base learner. It operates on weights of the loss function terms. For a training set of N samples

and a given loss function L, the global loss is

$$Loss = \sum_{i=1}^{N} L(y_i, f(x_i))$$

Different ways of choosing loss function yield different boosting algorithm. The most famous algorithm for boosting is Adaboost (Freund & Schapire, 1997), which is especially set for binary classification problem and uses exponential loss.

In our case, the base learner is the decision tree. Adaboost trains the decision tree for several iterations, but with a different weighted training data for each iteration. That is, each time a decision tree is trained, the wrongly predicted data records in the current iteration will be assigned higher weights in the computation of the loss function for the next training iteration of the decision. The final output of Adaboost is a `sgn` function (*sign function*) of a weighted sum of all "learners" trained in the whole procedure (the decision tree with different weights vectors).

For a training set of N samples, the whole procedure for Adaboost is shown below (Murphy, 2012):

Initialize $\omega_i = 1/N$
for $i = 1$ to M **do**
  Fit the classifier $\phi_m(x)$ to the training set using weights w
  Compute $err_m = \frac{\sum_{i=1}^{N} \omega_i I(\tilde{y_i} \neq \phi_m(x_i))}{\sum_{i=1}^{N} \omega_i}$
  Compute $\alpha_m = \log[(1 - err_m)/err_m]$
  set $\omega_i \leftarrow \omega_i \exp[\alpha_m I(\tilde{y_i} \neq \phi_m(x_i)]$
end for
return $f(x) = \text{sgn}(\sum_{m=1}^{M} \alpha_m \phi_m(x))$

In which M is the number of iterations (10 in our experiment), ω_i is the weight for i-th data, $I(\cdot)$ is the indicator function, $\tilde{y_i} \in \{1, -1\}$ is the class label of training data, and $\phi_m(x)$ is the decision tree model in our case.

Boosting has had stunning empirical success (Caruana & Niculescu-Mizil, 2006). More detailed explanation and analysis of boosting can be found in Bühlmann and Hothorn (2007) and Hastie, Tibshirani, and Friedman (2009). The Adaboost algorithm was implemented in this experiment to improve the results obtained by Desmarais et al. (2015). The results are reported in section 7.

5. METHODOLOGY AND PERFORMANCE CRITERION

To estimate the ability of an algorithm to validate a Q-matrix, we perturbate a "correct" Q-matrix and verify if the algorithm is able to recover this correct matrix by identifying the cells that were perturbated while avoiding to classify unperturbated cells as perturbated. In this experiment, only one perturbation is introduced. For synthetic data, the "correct" matrix is known and is the one used in the generation of the data. For real data, we assume the expert's is the correct one, albeit it may contain errors.

Table 1: Q-matrix for validation

Name	Number of			Description
	Skills	Items	Cases	
QM1	3	11	536	Expert driven from (Henson, Templin, & Willse, 2009)
QM2	3	11	536	Expert driven from (De La Torre, 2008)
QM3	5	11	536	Expert driven from (Robitzsch, Kiefer, George, & Uenlue, 2015)
QM4	3	11	536	Data driven, SVD based

In order to use a standard performance measure, we define the following categories of correct and incorrect classifications as the number of:

- **True Positives (TP)**: perturbed cell correctly recovered
- **True Negatives (TN)**: non perturbed cell left unchanged
- **False Positives (FP)**: non perturbed cell incorrectly recovered
- **False Negatives (FN)**: perturbed cell left unchanged

We give equal weight to perturbed and unperturbed cells and use the F1-score, or F-score for short. The F-score is defined as

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

In which *precision* is calculated by the model accuracy on non-perturbated cell while *recall* is calculated by the model accuracy on perturbated cell.

6. DATASET

For the sake of comparison, we use the same datasets as the ones used in Desmarais et al. (2015). Table 1 provides the basic information and source of each dataset.

7. RESULT

The results of applying Adaboost over the decision tree (DT) are reported in table 2 for synthetic data and Table 3 for real data. The individual results of each algorithm are reported (minRSS, maxDiff, and ALSC), along with the decision tree (DT) and the boosted decision tree (BDT). Different improvement over baselines are reported as:

- **DT %Gain**: the Decision Tree (DT) improvement over the **best** of the three individual algorithm (minRSS, maxDiff, ALSC)
- **BDT %Gain**: Boosted Decision Tree improvement over the DT performance, which corresponds to the gain we get from boosting.

Let us focus on the F-Score which is the most informative since it combines results of the perturbed and non perturbed

Table 2: Results for synthetic data

QM	Individual: minRSS	Individual: maxDiff	Individual: ALSC	Ensemble: DT (%Gain)	Ensemble: BDT (%Gain)
Accuracy of perturbated cells					
1	0.809	0.465	0.825	0.946 (69.4%)	**0.951** (9.2%)
2	0.069	0.259	0.359	0.828 (73.2%)	**0.903** (43.5%)
3	0.961	0.488	0.953	**1.000** (99.7%)	**1.000** (0.0%)
4	0.903	0.489	0.853	0.956 (54.3%)	**0.971** (33.9%)
$\overline{\mathbf{X}}$	0.685	0.425	0.747	0.933 (74.2%)	**0.956** (21.7%)
Accuracy of non perturbated cells					
1	0.970	0.558	0.387	**0.990** (65.1%)	**0.990** (0.0%)
2	0.987	0.529	0.431	0.989 (20.5%)	**0.996** (59.1%)
3	0.950	0.258	0.736	0.994 (88.9%)	**1.000** (100.0%)
4	0.966	0.559	0.391	0.997 (92.2%)	**0.998** (19.2%)
$\overline{\mathbf{X}}$	0.968	0.476	0.486	0.993 (65.3%)	**0.996** (49.4%)
F-score					
1	0.882	0.507	0.527	0.968 (72.4%)	**0.970** (7.4%)
2	0.128	0.348	0.392	0.902 (83.8%)	**0.947** (46.1%)
3	0.955	0.337	0.831	0.997 (93.5%)	**1.000** (100.0%)
4	0.934	0.522	0.536	0.976 (64.0%)	**0.984** (33.6%)
$\overline{\mathbf{X}}$	0.725	0.429	0.571	0.961 (78.4%)	**0.975** (46.4%)

Table 3: Results for real data

QM	Individual: minRSS	Individual: maxDiff	Individual: ALSC	Ensemble: DT (%Gain)	Ensemble: BDT (%Gain)
Accuracy of perturbated cells					
1	0.485	0.167	0.515	**0.758** (50.0%)	**0.758** (0.0%)
2	0.345	0.093	0.564	0.618 (12.5%)	**0.764** (38.1%)
3	0.212	0.091	0.364	**0.818** (71.4%)	**0.818** (0.0%)
4	0.394	0.111	0.576	0.576 (0.0%)	**0.818** (57.1%)
$\overline{\mathbf{X}}$	0.359	0.115	0.505	0.692 (33.5%)	**0.789** (23.8%)
Accuracy of non perturbated cells					
1	0.435	**0.670**	0.418	0.606 (−19.4%)	0.606 (0.0%)
2	0.875	0.929	0.110	0.956 (37.9%)	**0.966** (21.4%)
3	0.661	**0.830**	0.219	0.785 (−26.2%)	0.752 (−15.1%)
4	0.520	**0.889**	0.148	0.546 (−308.7%)	0.658 (24.7%)
$\overline{\mathbf{X}}$	0.623	**0.829**	0.224	0.723 (−79.1%)	0.746 (8.0%)
F-score					
1	0.459	0.267	0.461	**0.673** (39.4%)	**0.673** (0.0%)
2	0.495	0.168	0.184	0.751 (50.6%)	**0.853** (40.9%)
3	0.321	0.164	0.273	**0.801** (70.7%)	0.784 (−8.7%)
4	0.448	0.198	0.235	0.560 (20.3%)	**0.730** (38.5%)
$\overline{\mathbf{X}}$	0.431	0.199	0.288	0.696 (45.25%)	**0.760** (17.8%)

cells of the Q-matrix. For synthetic data, the error reduction of boosting over the gain from the decision tree is substantially improved for all Q-matrices. The range of improvement is from 7% to 100%. For real data, two of the four Q-matrices show substantial improvements of around 40%, whereas the other two show no improvements, even a decrease of 8.7% for Q-matrix 3 which is characterized by a single skill per item. However, let us recall that we assume the expert Q-matrices are correct, which may be over optimistic. Violation of this assumption could negatively affect some of the Q-matrices scores for real data.

Note that QM3 has an inconsistent 100% gain from boosting with synthetic data compared to a small loss is obtained with real data. The value of 100% should be taken cautiously because the F-score difference is measured close to the boundary of 1 and therefore the result of only a few cases in our sample. Nevertheless, the fact that a very high F-score is obtained for synthetic data compared to real data does raise attention and might be related to the fact that it is the only single skill per item matrix.

8. DISCUSSION

This study shows that the gain obtained from combining the output of multiple Q-matrix refinement algorithms with a decision tree can be further improved with boosting. The results for synthetic data show an F-score error reduction from boosting over the DT score of close to 50% on average for all four Q-matrices, and a 18% reduction for real data. Compared with the score of the three individual refinement algorithms, minRSS, maxDiff, and ALSC, the combined ensemble learning of decision tree is very effective.

However, we find strong differences between the Q-matrices. For eg., QM2 benefits of improvements close to 50% (QM2), while QM1 has a null improvement for real data and only 7.4% for synthetic data. In that respect, the boosting does not provide a gain that is as systematic as the one obtained from the DT which is positive for all matrices.

An important advantage of the meta-learning approach outlined here is that it can apply to any combination of algorithms to validate Q-matrices. Future work could look into combining more than the three algorithms of this study, and add new algorithms that potentially outperform them. And if the current results generalize, we would expect to make supplementary gains over any of them.

Moreover, the Q-matrices used in this research are quite small in size. The performance of boosted decision tree on larger Q-matrix and larger dataset would also be of interest.

However, besides the Q-matrix-based algorithms mentioned above, there are other frameworks for knowledge tracing or domain modeling, especially when dealing with dynamic data. For example, there are Learning Factor Analysis (Cen, Koedinger, & Junker, 2006), Weighted CRP (Lindsey, Khajah, & Mozer, 2014), HMM-based Bayesian Knowledge Tracing (Corbett & Anderson, 1994; Lindsey et al., 2014) and other HMM-based models (González-Brenes, 2015). Comparison with these frameworks are also left to future work.

References

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Bühlmann, P., & Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 477–505.

Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on machine learning* (pp. 161–168).

Cen, H., Koedinger, K., & Junker, B. (2006). Learning factors analysis–a general method for cognitive model evaluation and improvement. In *Intelligent tutoring systems* (pp. 164–175).

Chen, Y., Liu, J., Xu, G., & Ying, Z. (2015). Statistical analysis of q-matrix based diagnostic classification models. *Journal of the American Statistical Association*, *110*(510), 850–866.

Chiu, C.-Y., & Douglas, J. (2013). A nonparametric approach to cognitive diagnosis by proximity to ideal response patterns. *Journal of Classification*, *30*(2), 225–250.

Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, *4*(4), 253–278.

De La Torre, J. (2008). An empirically based method of q-matrix validation for the dina model: Development and applications. *Journal of educational measurement*, *45*(4), 343–362.

De La Torre, J. (2009). Dina model and parameter estimation: A didactic. *Journal of Educational and Behavioral Statistics*, *34*(1), 115–130.

de la Torre, J., & Chiu, C.-Y. (2015). A general method of empirical q-matrix validation. *Psychometrika*, 1–21.

Desmarais, M. C., & Naceur, R. (2013). A matrix factorization method for mapping items to skills and for enhancing expert-based q-matrices. In *Artificial intelligence in education* (pp. 441–450).

Desmarais, M. C., Xu, P., & Beheshti, B. (2015). Combining techniques to refine item to skills q-matrices with a partition tree. In *Educational data mining 2015*.

Durand, G., Belacel, N., & Goutte, C. (2015). Evaluation of expert-based q-matrices predictive quality in matrix factorization models. In *Design for teaching and learning in a networked world* (pp. 56–69). Springer.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, *55*(1), 119–139.

González-Brenes, J. P. (2015). Modeling skill acquisition over time with sequence and topic modeling. In *Aistats*.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning*. Springer.

Henson, R. A., Templin, J. L., & Willse, J. T. (2009). Defining a family of cognitive diagnosis models using log-linear models with latent variables. *Psychometrika*, *74*(2), 191–210.

Lindsey, R. V., Khajah, M., & Mozer, M. C. (2014). Automatic discovery of cognitive skills to improve the prediction of student learning. In *Advances in neural information processing systems* (pp. 1386–1394).

Madison, M. J., & Bradshaw, L. P. (2015). The effects of q-matrix design on classification accuracy in the log-linear cognitive diagnosis model. *Educational and Psychological Measurement*, *75*(3), 491–511.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*(1), 81–106.

Quinlan, J. R. (1993). *C4. 5: Programs for machine learning* (Vol. 1). Morgan Kaufmann.

Robitzsch, A., Kiefer, T., George, A. C., & Uenlue, A. (2015). Cdm: Cognitive diagnosis modeling [Computer software manual]. Retrieved from `http://CRAN.R-project.org/package=CDM` (R package version 4.5-0)

Rupp, A. A., & Templin, J. (2008). The effects of q-matrix misspecification on parameter estimates and classification accuracy in the dina model. *Educational and Psychological Measurement*, *68*(1), 78–96.

Schapire, R. E., & Freund, Y. (2012). *Boosting: Foundations and algorithms*. MIT press.

Tatsuoka, K. K. (1983). Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of educational measurement*, *20*(4), 345–354.

Therneau, T. M., Atkinson, B., & Ripley, B. (2015). rpart: Recursive partitioning. r package version 4.1-10. *Computer software program retrieved from http://CRAN. R-project. org/package= rpart*.

Wang, S., & Douglas, J. (2015). Consistency of nonparametric classification in cognitive diagnosis. *Psychometrika*, *80*(1), 85–100.

Individualizing Bayesian Knowledge Tracing. Are Skill Parameters More Important Than Student Parameters?

Michael V. Yudelson
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213 USA
+1 (412) 268-5595
yudelson@cs.cmu.deu

ABSTRACT

Bayesian Knowledge Tracing (BKT) models were in active use in the Intelligent Tutoring Systems (ITS) field for over 20 years. They have been intensively studied, and a number of useful extensions to them were proposed and experimentally tested. Among the most widely researched extensions to BKT models are various types of individualization. Individualization, broadly defined, is a way to account for variability in students that are working with the ITS that uses BKT model to represent and track student learning. One of the approaches to individualizing BKT is to split its parameters into per-skill and per-student components. In this work, we are proposing an approach to individualizing BKT that is based on Hierarchical Bayesian Models (HBM) and, in addition to capturing student-level variability in the data, weighs the contribution of per-student and per-skill effects to the overall variance in the data.

Keywords

Student models of practice, Bayesian knowledge tracing, hierarchical Bayesian models, skill vs. student parameterization.

1. INTRODUCTION

Bayesian Knowledge Tracing (BKT) is one of the most popular student modeling techniques in the field of Intelligent Tutoring Systems (ITS). It has been in active use for over two decades and has been confirmed to be the modeling approach researchers can rely on.

Over the years, a large number of extensions to the standard BKT were proposed and tested in posthoc analyses as well as experimentally. Among the most widely researched additions to BKT is the ability to account for students' individual traits. It has been confirmed in the are of modeling student learning in general and in the case of BKT that accounting for student-level variability in the data could benefit the model's statistical goodness of fit, as well as potentially improve the generalizability of the model.

Known approaches could be separated into three groups. The first group, binary multiplexing of the initial skill mastery probability based on the student characteristics, for example, the correctness of the first response (Pardos & Heffernan, 2010). This method has been proven to benefit the overall student model quality, and the implementation of this approach was a runner-up in the 2010 KDD Cup data mining challenge. The second group, fitting BKT parameters not across students for a particular skill, but for a student/skill pair (Lee & Brunskill, 2012). This approach has not been evaluated for predictive correctness. The third group, are the methods separating BKT parameters into per-student and per-skill components (Corbett & Anderson, 1995; Yudelson et al., 2013). The two approaches from the third group were shown to improve model fits reliably.

While the BKT individualization approaches mentioned above were successful in one way or the other, are arguably yet to achieve a sufficient flexibility and rigor of the available parameterization devices. In this paper, we propose and investigate an individualized Bayesian Knowledge Tracing that, on top of refining certain aspects of its predecessor (Yudelson et al., 2013), draws on the flexibility of the Hierarchical Bayesian Models' representation to capture relative weight of student-level and skill-level variability in the learning data as defined by respective parameters. Also, we empirically explore the possibility of clustering student-level factors via mixes of Gaussian distributions.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 outlines the methods. Section 4 describes the data we used for this investigation. Section 5 talks about the results. Finally, Section 6 closes with a few discussion points.

2. RELATED WORK

2.1 Bayesian Knowledge Tracing

Bayesian Knowledge Tracing (BKT) is a probabilistic framework (Corbett & Anderson, 1995) it is used to assess student progress with a unit of knowledge often referred to as skill. Upon correct or incorrect action, an estimate of student mastery of skill(s) is re-computed. Computationally, BKT is a Hidden Markov Model with two hidden states, representing whether a particular skill is un-mastered or mastered. Observations of student performance on opportunities to practice a skill are binary: a student either solves a problem step correctly or not (due to error or because of a hint request). While students might go through dozens of attempts to get a particular step correct, traditionally, only students' first attempts are considered for updating skill mastery estimates.

There are four skill parameters used in BKT: initial probability of knowing the skill a priori – $p(L_0)$ (or *p-init*), probability of student's knowledge of a skill transitioning from not known to known state after an opportunity to apply it – $p(T)$ (or *p-learn*), probability to make a mistake when applying a known skill – $p(S)$ (or *p-slip*), and probability of correctly applying a not-known skill – $p(G)$ (or *p-guess*). Given that parameters are set for all skills, the formulae used to update student knowledge of skills are as follows. The initial probability of student u mastering skill *k* is set to the *p-init* parameter for that skill Equation (1a). Depending on whether the student u applied skill k correctly or incorrectly, the conditional probability is computed either using Equation (1b) or Equation (1c). The conditional probability is used to update the

probability of skill mastery according to Equation (1d). To compute the probability of student *u* applying the skill *k* correctly on an upcoming practice opportunity one uses Equation (1e).

$$p(L_1)_u^k = p(L_0)^k \quad (1a)$$

$$p(L_{t+1}|obs = correct)_u^k = \frac{p(L_t)_u^k \cdot (1 - p(S)^k)}{p(L_t)_u^k \cdot (1 - p(S)^k) + (1 - p(L_t)_u^k) \cdot p(G)^k} \quad (1b)$$

$$p(L_{t+1}|obs = wrong)_u^k = \frac{p(L_t)_u^k \cdot p(S)^k}{p(L_t)_u^k \cdot p(S)^k + (1 - p(L_t)_u^k) \cdot (1 - p(G)^k)} \quad (1c)$$

$$p(L_{t+1})_u^k = p(L_{t+1}|obs)_u^k + (1 - p(L_{t+1}|obs)_u^k) \cdot p(T)^k \quad (1d)$$

$$p(C_{t+1})_u^k = p(L_t)_u^k \cdot (1 - p(G)^k) + (1 - p(L_t)_u^k) \cdot p(G)^k \quad (1e)$$

2.2 Introducing Student-Level Factors to the Bayesian Knowledge Tracing

Having student-level parameters is a regular feature of models of student learning and learning performance. The logistic regression based Rasch model (van der Linden & Hambleton, 1997) that captures test item complexity and its extension –the Additive Factors Model (Cen et al., 2008) both include a parameter to account for variability in the student a priori abilities. Including student-level parameters in these models helps both the fit as well as the interpretability of the models overall.

There were a few attempts to introduce student-specific parameters to otherwise skill-only standard BKY. The original work on BKT (Corbett & Anderson, 1995) discussed fitting skill-level and student-level parameters on respective slices of the data to later combine and apply the two in the context of each student-skill pair. As a result, the correlation of expected and observed within-student accuracies was higher for the thus individualized model.

Another approach to individualization suggests the multiplexing probability of initial skill mastery (*p-init*) based on student cohort (Pardos & Heffernan, 2010). Based on the correctness of the first student's response, the appropriate skill *p-init* is set to the lower or higher predetermined constant. This prior-per-student model outperforms standard BKT on a significant fraction of problem sets authors considered.

According to yet another approach (Lee & Brunskill, 2012), BKT parameters were fit within each student-skill pair's data slice and not across skills or students. Authors did not discuss on the goodness of fit of their individualized models, however. Their primary focus was on whether the individualized model when deployed in an intelligent tutoring system, would schedule fewer or more problems to be solved as compared to standard BKT model. The conclusion was that a considerable fraction of students, as judged by individualized model, would have received a significantly different amount of practice problems.

Finally, another individualization approach that we would be using for comparison in this work suggests something akin to the original discussion of the BKT individualization (Yudelson et al., 2013). Student and skill components of BKT parameters are fit one set after the other using a coordinate gradient descent procedure with an active parameter set maintained throughout the process. In addition to improved fits, BKT models individualized this way were shown to lead to optimized problem-sequences leading to saving students some efforts.

Overall, there is enough evidence that introducing student-level parameters to BKT benefits the fit of the model and could optimize student learning experience.

2.3 Introducing Item-Level Factors to the Bayesian Knowledge Tracing

Recently, a noticeable amount of work focused on addressing item-level variability in BKT models. Pardos & Heffernan (2011) presented their KT-IDEM model that features special nodes that capture item difficulties and, together with skill-level latent variables are influencing the student performance.

In the approach Huang and colleagues took (Huang et al., 2015), it is possible to address not just items, but even item level features, adding parameters in a way it is done in regression analysis. In another work (Khajah et al., 2014), authors are discussing merging an IRT model and BKT model. This approach resulted in an HBM that combines features of both. It is worth to note that the latter two use Markov Chain Monte Carlo methods to fit their models.

3. METHODS

Our objective is to introduce further improvements to the approach to individualizing BKT and draw comparisons to regular BKT as well its original version in terms of statistical fitness as well as and to attempt to judge the plausibility of their respective student-level parameters.

3.1 Individualized BKT Model via Parameter-Splitting

Individualization of the BKT that was proposed in (Yudelson et al., 2013) prescribes to put every individualized parameter in the context of a particular student that works on a particular skill. In this context, *p-init*, *p-learn*, *p-slip*, and *p-guess* parameters have two components: a per-skill component and a per-student component. The two are combined using a pairing function shown in Equation 2a. Here, components are first converted from probability scale to log-odds scale using logit function (Equation 2b), added, and the sum is converted back to the probability scale using sigmoid function (Equation 2c). An individualized model, where all per-student components are equal to 0.5 (0 on the log-odds scale) is equivalent to the standard BKT model.

$$f(P_k^i, P_u^i) = S\left(l(P_k^i) + l(P_u^i)\right) \quad (2a)$$

$$l(p) = \ln\left(\frac{p}{1-p}\right) \quad (2b)$$

$$\mathrm{S(x)} = \frac{1}{1 + \mathrm{e}^{-\mathrm{x}}} \quad (2c)$$

Fitting of such individualized BKT (iBKT) model is done by computing gradients of the log-likelihood function given individual student/skill data samples with respect to every iBKT parameter (Levinson et al., 1983). On every odd run, gradients are aggregated across skills to update skill component of the parameters. On every even run, the gradients are aggregated across students to update respective student components. This

block-coordinate descent is performed until all parameter values stabilize up to a pre-set tolerance criterion. An active set of parameter components is maintained to fit only those that still haven't stabilized. An extended discussion of the method, as well as derived formulas for the gradients is given in the original publication of this work (Yudelson et al., 2013).

The standard and individualized model described above we implemented in the tool called `hmm-scalable`. The tool has a suite of solvers, including the classical BKT Expectation Maximization solver for standard BKT, as well as a set of stochastic and conjugate gradient descent solvers. `Hmm-scalable` is freely available on GitHub repository[1] of the International Educational Data Mining Society (standard BKT models only).

3.2 Individualized BKT via Hierarchical Bayesian Model

We have also implemented the BKT as well as the iBKT approach described above in the form of a Hierarchical Bayesian Model (HBM). HBMs allow for a more universal and flexible way of representing iBKT. The HMB BKT just like the hmm-scalable BKT had 4*N* parameters, where *N* is the number of skills. In the iBKT models, both hmm-scalable and HBM version, only the *p-init* and *p-learn* were individualized. Thus, the number of parameters in the hmm-scalable version of iBKT was 4*N*+2*M*, where *M* is the number of students. HBM version of the iBKT treated per-student parameters as being drawn from Gaussian distributions and had 4 hyper-parameters: mean and standard deviation for student-level *p-init* and *p-learn*. While we did not specifically check or prove this, but intuitively, confining a parameter to the bounds of a particular distribution serves as a form of regularization and, theoretically, could improve the generalizability of the model. Although iBKT models 4*N*+2*M* had parameters, the per-student and per-skill parameters, when combined using the pairing function from Equation 2a, could result in up to 2*N*+2*NM* in-context parameters. *P-guess* and *p-slip* were not individualized (2*N*), 2NM represents all possible combinations of students and skills for *p-init* and *p-learn*.

$$f\left(P_k^i, P_u^i, W_0, W_k, W_u, W_{uk}\right) = \\ = S\left(W_0 + W_k l\left(P_k^i\right) + W_u l\left(P_u^i\right) + W_{uk} l\left(P_u^i\right) l\left(P_k^i\right)\right) \quad (3)$$

The main contribution of this paper is to not only mix per-student and per-skill parameters together but to weight each component of the mixture in an attempt to define whether either one has a larger impact on the resulting in-the-context parameter value. We have taken Equation (2a) and changed into Equation (3). Here we have the bias term (W_0), the weights for the per-skill and per-student components (W_k and W_u respectively), and also the interaction term for the two with the weight (W_{uk}). The $W_{\cdot}$ weights are drawn from Gaussian distribution. Each of them is constrained to [0, 1], and the sum is fixed at 2. We have used the same $W_{\cdot}$ weights for mixing both *p-init* and *p-learn*. Thus, we have 8 additional hyperparameters and this new model, that we will refer to as iBKT-W HBM, has 4*N*+2*M*+4 parameters and 12 hyper-parameters. If $\{W_0, W_k, W_g, W_{gk}\}$ weights were set to {0, 1, 1, 0} respectively, the model would we equivalent to the iBKT HBM model.

When exploring the per-student parameter values if the iBKT-W HBM model, we have noticed that, in spite of being drawn from the Gaussian distribution, the actual distribution has a hint of being binomial (rf. Figure 1). It is especially visible for the distribution of the per-student values of *p-init*. In order to address this phenomenon, we have created yet another HBM model, that we will call iBKT-W-2G HBM, where the per-student *p-init* and *p-learn* parameters will be drawn from a mixture of 2 Gaussian distributions. In this new model, there are 4 means of the Gaussians distributions (2 for per-student *p-init* and 2 per-student for *p-learn*), 2 variances (1 for per-student *p-init* and 1 per-student for *p-learn*) instead of 4 as in iBKT-W HBM. The membership in one or the other mixture is modeled by a 2-parameters categorical distribution based on Dirichlet(1,1) distribution. Thus, there are, just as before, 4*N*+2*M*+4 parameters, while the number of hyperparameters is 16. Table 1 summarizes the information about parameters of all of the models we have considered in this work.

HBM versions of the three iBKT models are not supported by `hmm-scalable`. To build them we used BUGS language (Lunn et al., 2009) implemented as `rjags` package in `R` (Plummer, 2016). As opposed to `hmm-scalable`, that uses a form of exact inference, BUGS models were build using the Gibbs Sampler implemented in the `rjags` package.

To fit HBM iBKT models we used 10 chains running in parallel for the duration of 500 iterations. Unfortunately, it is not possible whether a model fit using a Gibbs sampler has converged. It is, however, possible to say whether it did not. In our experimental runs, we have confirmed there were no signs that the models failed to converge. Each model took roughly 1 hour to finish.

Table 1. Model parameters and hyper-parameters. Number of skills – *N*, number of students – *M*

Model	Parameters	Hyper-parameters
Majority Class	0	0
Standard BKT hmm-scalable	4N	0
Standard BKT JAGS	4N	0
iBKT hmm-scalable*	4N+2M	0
iBKT HBM*	4N+2M	4
iBKT-W HBM *	4N+2M+4	12
iBKT-W-2G HBM *	4N+2M+4	16

* for all iBKT models we only individualize *p-init* and *p-learn*.

4. DATA

We used the data from the KDD Cup 2010 Educational Datamining Challenge[2]. The data was donated by Carnegie Learning Inc., a publisher of mathematics curricula and a producer of intelligent tutoring system – Carnegie Learning's Cognitive Tutor – for middle school, high school, and college. The KDD Cup 2010 datasets are quite large. Algebra dataset has close to 10 million student transactions, and pre-algebra dataset has a little over 20 million transactions.

Although computational capabilities of the `hmm-scalable` tool allow fitting BKT and iBKT models within minutes, R

[1] `https://github.com/IEDMS/standard-bkt`

[2] `http://pslcdatashop.web.cmu.edu/KDDCup`

implementation of the Gibbs Sampler and the BUGS language are not as scalable. Because of that, we have selected a subset of the pre-algebra dataset, namely, a sample where students worked on Linear Inequalities unit. This sample consisted of 66,307 transactions of 336 students. This sample only contained transactions labeled with the skills that the Carnegie Learning's Cognitive Tutor tracks. There were 30 skills that the unit on linear inequalities taught.

From the rich feature set of the data we took four columns: success at first attempt at a problem step (student activity is blocked and sequenced into working on individual problem steps and BKT traditionally only looks at the first attempt; anonymous student id; concatenation of curriculum unit, section, and problem (was not necessary for our analyses, but required by `hmm-scalable`); and relevant skill(s) practiced at that particular step.

5. RESULTS

5.1 Model Fits

The results of statistical fitness of the models we have discussed are in Table 2. There we list four fitness metrics, the Deviance Information Criterion (van der Linde, 2005), root mean squared error, Accuracy and area under ROC curve (A'). DIC is a metric based on log-likelihood. It is often used for Bayesian model selection. Accuracy is a point measure of how often the model guesses the correct response (here whether the student was correct or incorrect). RMSE goes a little further by quantifying how close the each prediction is to the correct classification of a correct or incorrect response. The area under the ROC curve is a measure of how well the model can tell the classes or responses apart. As the name suggests, it is a curve metric, without a working point, like accuracy (with which a 0.5 threshold is often used).

As wee can see in Table 2, the majority class model performance is low as expected A' is at 0.50 (as it should be), accuracy is about 72%. There are usually more correct responses in the Carnegie Learning's Cognitive Tutor data since the tutor breaks problems into steps and guides students towards the correct solution.

As we move down in Table 2, we can see that model accuracies start improving. Standard BKT models outperform Majority Class. There is a small advantage of the HBM model fit using `R` implementation of JAGS over the `hmm-scalable`. iBKT models (here we only individualize *p-init* and *p-learn*) are a further improvement of the fit, again, with a small advantage for the HBM version of the model. The weighted version of the iBKT (iBKT-W) is only implemented as an HBM and, again, shows an improvement overall (in terms of DIC, RMSE, and A').

Table 2. Performance of the models

Model	DIC	RMSE	Acc.	A'
Majority Class		0.52516	0.7242	0.5000
Standard BKT hmm-scalable	66230	0.40571	0.7561	0.7649
Standard BKT HBM	65347	0.40299	0.7569	0.7728
iBKT hmm-scalable*	64215	0.39376	0.7680	0.7990
iBKT HBM *	63644	0.39287	0.7692	0.7992
iBKT-W HBM *	63587	0.39236	0.7687	0.8005
iBKT-W-2G HBM *	63412	0.39252	0.7689	0.8005

* for all iBKT models we only individualize *p-init* and *p-learn*.

In addition to observing model fits, we have performed one round of 3-fold item-stratified cross-validation to verify whether the differences between the iBKT model fit by `hmm-scalable` and the iBKT-W model fit by JAGS become more visible. Although the fit metrics deteriorated a bit, the partial order of the models regarding the goodness of fit did not change.

5.2 Per-Skill and Per-Student Parameters

When we plotted the densities of per-student *p-init* and *p-learn* parameters for the weighted iBKT, we have noticed that the distributions had a hint of bimodality, especially the distribution of per-student *p-init* (rf. Figure 1). Given that the HBM is drawing parameter values from a Gaussian distribution, the bi-modality is quite pronounced. To check our intuition, we have constructed a modified version of the weighted iBKT where per-student p-init and p-learn are mixstures of two Gaussians. The new model, iBKT-W-2G, did not show improvement in fit statistics, except for DIC. However, the distributions of the corresponding per-student *p-init* and *p-learn* were visibly bimodal (rf. Figure 6). The two means for the p-init parameters are 0.280 and 0.786. The two means for the p-learn parameters are 0.277 and 0.630.

The weights for pairing the per-student and per-skill parameters for both of the weighted iBKT models are given in Table 3. Both the bias weight W_0 and interaction W_{uk} seem to be sufficiently small. Although there is no exact agreement between the two models, in both the weight of the per-skill parameters (W_k) are two to three times smaller than that of per-student parameters (W_u).

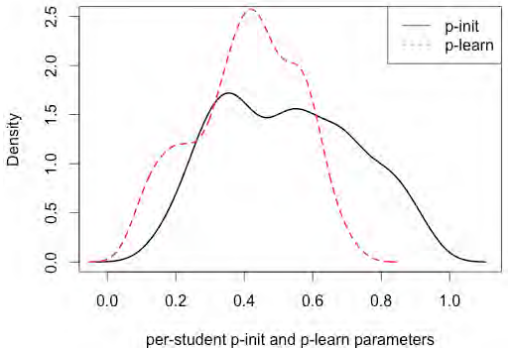


Figure 1. Density plots for per-student *p-init* and *p-learn* parameters of iBKT-W HBM model.

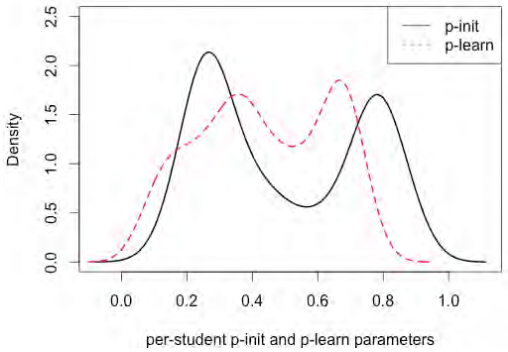


Figure 2. Density plots for per-student *p-init* and *p-learn* parameters of iBKT-W-2G HBM model.

Table 3. Skill-student weights in iBKT-W models

Model	W_0	W_k	W_u	W_{uk}

iBKT-W HB	0.012	0.565	1.420	0.004
iBKT-W-2G HBM	0.019	0.700	1.274	0.007

5.3 Extra Look At Per-Skill and Per-Student Parameters

In an attempt to investigate the differences between iBKT model fit using `hmm-scalable` and the iBKT-W-2G fit using JAGS, we have plotted the per-student *p-init* and *p-learn* parameters for both. The respective plots are in Figure 3 and Figure 4. As we can see in Figure 3, where per-student parameters of iBKT `hmm-scalable` model are plotted, correlation of *p-init* and *p-learn* is mid-range and is equal 0.55. Notably, a tangible portion of students, as estimated by the model, have low *p-init* and high *p-learn* parameters. If we interpret *p-init* as student's overall prior preparation and *p-learn* as student's overall rate of learning, these would be the students that came in with the low level of knowledge and quickly caught up. Using the same logic, there are also a few students that came in with high prior knowledge but suffered from low learning rate.

The plot of per-student *p-init* and *p-learn* parameters of iBKT-W-2G HBM model is entirely different (rf. Figure 8). The correlation is very high – 0.90. Although the student points are lined up almost linearly, it is possible to discern two clusters (lower left, and upper right) that roughly correspond to two mixed Gaussians represented by a categorical node in the model. Here, there are effectively no students in the upper left or bottom right corners of the graph. Namely, those arriving with lower preparation, but the high rate of learning, or, vice-versa, high preparation, but the lower rate of learning. The former is unfortunate since the unprepared students that can quickly close the gap are, arguably, the most desired ones since they make the application that assisted them (e.g., Carnegie Learning's Cognitive Tutor) shine.

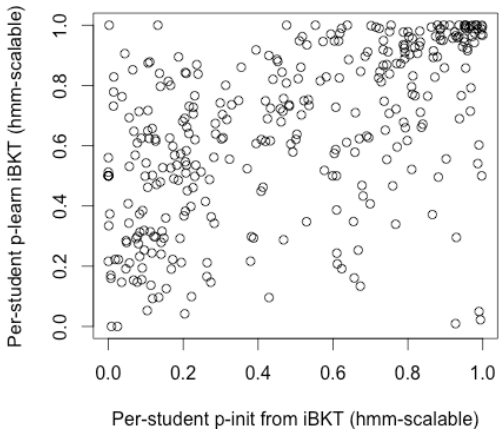


Figure 3. Scatter plot of per-student *p-init* (x-axis) and *p-learn* (y-axis) from iBKT model fit by hmm-scalable. The correlation between the two is 0.55 (significant at 0.001 level).

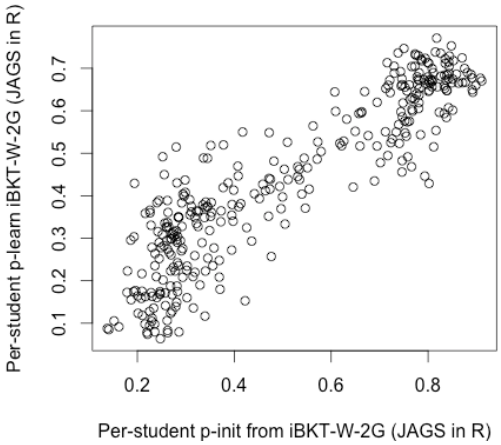


Figure 4. Scatter plot of per-student *p-init* (x-axis) and *p-learn* (y-axis) from iBKT-W-2G model fit by JAGS in R. The correlation between the two is 0.90 (significant at 0.001 level).

6. DISCUSSION

6.1 Small Differences in Statistical Fits

Arguably the most pressing question about comparing the `hmm-scalable`-fit iBKT model and the HBM models is why the differences in statistical accuracy are so small. Given that some of the changes in per-student parameters are quite large (rf. Figures 3 and 4), we are to expect more pronounced differentiation, especially since the fitting method and parameterization changed.

We would like to refer to an earlier work where we examined alternative parameterizations of a logistic regression model of student math learning (Yudelson et al., 2011). As we have found there, despite virtually no difference in statistical fit, the parameter values and especially their interpretability improved. We did not estimate the interpretability of the parameter values of the HBM models, however, the relative distribution of the iBKT-W-2G HBM per-student parameters is, arguably, more realistic than that of the iBKT `hmm-scalable`.

Besides, as we were able to show in (Yudelson & Ritter, 2015), the absence of a *tangible* difference in statistical fit between two models may, none the less, correspond to considerable variance in assigned practice when the models compared are deployed in the actual system and used for knowledge tracking and problem selection.

6.2 What Do The Gaussians Mixtures Represent?

We have followed the trace of the possible bi-modal distributions of per-student *p-init* and *p-learn* parameters in the iBKT-W and constructed iBKT-W-2G model where per-student parameters are represented as mixtures of 2 Gaussian distributions with the same standard deviation.

To reverse-engineer the fuzzy mixture variable that *clusters* students we have attempted to correlate it with a set of student performance metrics. These included: overall number of problems solved, time spent, hints requested (both on the first attempt at a step and overall), errors committed (both on the first attempt at a step and overall), percent correct (both on the first attempt at a step and overall), time spent per problem, errors committed and hints requested per problem. None of them correlated with the fuzzy mixture variable reliably. It is likely that the resulting

clustering represents some latent student factor, we just could not interpret it.

6.3 Weighting Per-Skill and Per-Student Parameters

We have tried more models than the two HBM iBKT-W's we reported. The models included those individualizing *p-init* and *p-learn* separately or together, with weighting or without, mixing 1, 2, or 3 Gaussians (18 variants overall) – in all cases per-student parameter component weight was two-to-three times larger than that of per-skill components. One explanation for that could be possible over-fitting. There are 336 students and 30 skills. Even though the model is hierarchical and both per-skill and per-student parameter values are regularized, they are an order of magnitude more per-student values. To confirm or disconfirm the over-fitting hypothesis we would have to perform multiple sample-and-fit rounds where the number of students is equal to the number of skills.

7. ACKNOWLEDGMENTS

The author would like to give special thanks to Mr. Christopher MacLellan for introducing him to the BUGS language, Dr. Ilya Goldin for sharing his draft of a single-skill BKT implementation in BUGS, and Dr. Kenneth R. Koedinger for useful feedback while this work took shape.

8. REFERENCES

[1] Cen, H., Koedinger, K.R., Junker, B.: Comparing Two IRT Models for Conjunctive Skills. In: Woolf, B.P., A ̈imeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 796–798. Springer, Heidelberg (2008)

[2] Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction 4(4), 253–278 (1995)

[3] Fawcett, T. (2006) An introduction to ROC analysis. Pattern Recognition Letters, 27, 861–874.

[4] Huang, Y., Gonzalez-Brenes, J. P., and Brusilovsky, P. (2015) The FAST toolkit for Unsupervised Learning of HMMs with Features. In: The Machine Learning Open Source Software Workshop at the 32nd International Conference on Machine Learning (ICML-MLOSS 2015).

[5] Khajah, M., Wing, R. M., Lindsey, R. V., & Mozer, M. C. (2014) Incorporating latent factors into knowledge tracing to predict individual differences in learning. In J. Stamper, Z. Pardos, M. Mavrikis, & B. M. McLaren (Eds), Proceedings of the 7th International Conference on Educational Data Mining (pp. 99-106).

[6] Lee, J.I., Brunskill, E.: The Impact on Individualizing Student Models on Necessary Practice Opportunities. In: Yacef, K., Zaïane, O.R., Hershkovitz, A., Yudelson, M., Stamper, J.C. (eds.) Proceedings of the 5th International Conference on Educational Data Mining (EDM 2012), pp. 118–125 (2012)

[7] Levinson, S.E., Rabiner, L.R., Sondhi, M.M.: An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition. Bell System Technical Journal 62(4), 1035–1074 (1983)

[8] van der Linde, A. (2005). DIC in variable selection. *Statistica Neerlandica*, 59: 45-56.

[9] van der Linden, W.J., Hambleton, R.K.: Handbook of Modern Item Response Theory. Springer, New York (1997)

[10] Lunn D, Spiegelhalter D, Thomas A, Best N. (2009) The BUGS project: Evolution, critique, and future directions. *Statistics in Medicine*, 28:3049-67.

[11] Pardos, Z.A., Heffernan, N.T.: Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 255–266. Springer, Heidelberg (2010)

[12] Pardos, Z. & Heffernan, N. (2011) KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. In Konstant et al. (eds.) Proceedings of the 20th International Conference on User Modeling, Adaptation and Personalization (UMAP). (pp. 243-254), Girona, Spain. Springer.

[13] Plummer, M. (2016). rjags: Bayesian Graphical Models using MCMC. R package version 4-5. https://CRAN.R-project.org/package=rjags

[14] Yudelson, M., Koedinger, K., Gordon, G. (2013) Individualized Bayesian Knowledge Tracing Models. In: Lane, H.C., and Yacef, K., Mostow, J., Pavlik, P.I. (eds.) Proceedings of 16th International Conference on Artificial Intelligence in Education (AIED 2013), Memphis, TN. LNCS vol. 7926, (pp. 171–180).

[15] Yudelson, M., Pavlik, P.I., Koedeinger, K.R. (2011) User Modeling – a Notoriously Black Art. In J.A. Konstan, R. Conejo, J.L. Marzo, and N. Oliver (Eds.) Proceedings of the 19th International Conference on User Modeling Adaptation and Personalization (UMAP 2011), Girona, Spain, (pp. 317-328).

[16] Yudelson, M. & Ritter, S. (2015) Small Improvement for the Model Accuracy – Big Difference for the Students. In: Industry Track Proceedings of 17th International Conference on Artificial Intelligence in Education (AIED 2015), Madrid, Spain.

Deep Learning + Student Modeling + Clustering: a Recipe for Effective Automatic Short Answer Grading

Yuan Zhang, Rajat Shah, and Min Chi
North Carolina State University
{yzhang93, rshah6, mchi}@ncsu.edu

ABSTRACT

In this work we tackled the task of Automatic Short Answer Grading (ASAG). While conventional ASAG research makes prediction mainly based on student answers referred as Answer-based, we leveraged the information about questions and student models into consideration. More specifically, we explore the Answer-based, Question, and Student models individually, and subsequently in various combined and composite models through feature engineering. Additionally, we extend the exploration of machine learning methods by utilizing Deep Belief Networks (DBN) together with other five classic classifiers. Our experimental results show that our proposed feature engineering models significantly out-performed the conventional Answer-based model and among the six machine learning classifiers, DBN is the best followed by SVM, and Naive Bayes is the worst.

1. INTRODUCTION

Developing effective Computer-based assessment has been increasingly gaining its importance over years and it is widely believed that open-ended problems are more effective to access student knowledge than multiple choices. The former require students to generate free text and communicate their responses and thus student answers are relatively immune to test-taking shortcuts like eliminating improbable answers. On the other hand, grading student's free text answers is often time-consuming and challenging. Therefore, much research has focused on how to automatically grade student free text answers. Generally speaking, research to date has concentrated on two sub-tasks: grading student essays, which includes checking the style, grammar, and coherence of an essay [13], and grading student short answers [16, 18, 19], which is the focus of this work. More formally, [7] defined short answers as those: 1) in the form of natural language; 2) requiring students to recall external knowledge that is not provided by the question; 3) of which the length ranges between one phrase to one paragraph; 4) focusing on the correctness of the content rather than the style; and 5) and are closed, which means that the answers have to match the specific facts corresponding to questions. The goal of this work is to explore effectiveness of various Machine Learning (ML) approaches on Automatic Short Answers Grading (ASAG). An ASAG system is one that automatically classify student answers into, correct or incorrect, based on the referred correct one(s).

Much of the prior research on ASAG is answer-based which involves applying various Natural Language Processing (NLP) techniques to extract a wide variety of text-based features directly from student answers. These features include various measurements of text similarities between student answers and the referred correct ones. Often time, the shorter the student answers, the harder to classify them into correct or incorrect because the limited text provides fewer lexical features. Many classic NLP approaches such as bag-of-words or keyword matching often fail to work. For example, Table 1 shows an example of student short answer extracted from our training corpus. In this example, using text similarities alone would fail to recognize that the student's answer is correct because it looks quite different from the referred correct answer.

Table 1: An Example of Student Short Answer.

Tutor: Why are there no potential energies involved in this problem?
Student: There is no second object that is massive and can have gravitational energy. **(Correct)**
Correct Answer: Because the rock is the only object in the system, there are no potential energies involved.

On the other hand, information about question and student knowledge can be handily used to improve the effectiveness of existing answer-based ASAG model. For example, in the example above if we know that the question is about "potential energy" and the student's knowledge on "potential energy" is very high, it is more likely that the student will answer the question correctly even though his/her answer looks quite different from the correct one. Thus in this paper we will investigate whether the effectiveness of ASAG can be further improved if we leverage question model, student model, or both into the answer-based model. To the best of our knowledge, this is the first comprehensive study exploring the effectiveness of feature space from all three models on the task of ASAG. For simplicity reasons, in the following we will refer the three models as Answer(Ans), Question(Ques), and Student (Stu) models respectively.

Prior research on ASAG has explored several classic ML classifiers such as Naïve Bayes and Decision Tree. In re-

cent years, Deep Belief Network (DBN) [5] has been successfully implemented and applied in a wide variety of real-world tasks [15,17]. DBN enables the automatic extraction of representative features via an unsupervised pre-training and it can learn the latent complex relationship among features. Given the potential complex connections among the features from Ans, Ques and Stu models, we investigated on leveraging DBN to exploit the more discriminative feature space to facilitate automatic grading. As far as we know, this is the first study to apply DBN to the task of ASAG.

To summarize, we investigated on improving ASAG by utilizing DBN together with five classic ML methods and by extending existing answer-based approaches to leverage a wide range of state features which are either based on or generated from Ans, Ques, and Stu models.

2. RELATED WORK

Popular Natural Language Tutors like AutoTutor [11] and BEETLE II [12] have extensively studied how to automatically understand student Natural Language inputs so that the system can respond to student's responses adaptively. Pulman and Sukkarieh used manually crafted patterns in the part-of-speech tagged answers for pattern matching with the correct answer [19]. Their approach is question-specific in that they applied Naïve Bayes and Decision Tree to automatically generate patterns for each question using a set of marked answers. Results showed their approach can achieve an average accuracy of 84%.

Mohler and Mihalcea developed an unsupervised approach using Knowledge-based and Corpus-based text-to-text similarity measures [18]. They used Latent Semantic Analysis coupled with domain specific corpus built from Wikipedia. Their resulted measures outperformed other similarity measures in that the former obtained Pearson correlation $r = 0.463$ between the computer assigned grades and average of human assigned grades.

Recently, Microsoft's Power Grading [2] took a semi-automated approach based on the observation that similar answers get similar grades. Thus, instead of directly grading student answers, Power Grading builds a hierarchy of short-answer clusters and lets human grader either grade the entire cluster with same score or manipulate the clusters as needed. Inspired by their work and promising results, we borrowed some of the features such as length and tf-idf from previous research into this work.

Our approach differed from previous research in that: 1) unlike relying solely on answer-based methods, we explored features from Ans, Ques and Stu models individually and combined; 2) our models are trained across all questions, that is, it is question-general instead of building question-specific classifiers in previous research; 3) previous approaches mainly involved two or three ML methods while we used a total of six including the state-of-the-art DBN together with five other traditional ML approaches.

3. METHODS

In this section, we will briefly describe the features involved in this study and the ML classifiers applied. For the latter, we will focus on DBN.

3.1 State Features

To investigate the impact of state features on the task of ASAG, we compare the effectiveness of various features from Ans, Ques and Stu models *individually* and *combined*. We also *composite* new features generated within or across different models.

3.1.1 Answer (Ans) Model

In [7], Burrows et al. identified two categories of answer-based approaches: corpus-based approaches are based on mapping the concepts in student answers to those in the reference correct answers [16], while alignment-based approaches are based on clustering student answers by some quality similarity estimates among student answer representations regardless of the correct answers. Our Ans model includes both corpus-based features and alignment-based ones.

Based on [2] and [18], we defined five Ans-based features by measuring the text similarity between student answer and the correct answer(s). The latter consist of the referred correct answer and the correct answers generated by students. More specifically, we have:

- *length difference*: the length difference (in words) between the student and the correct answers.
- *max-matched idf*: the maximum value of idf of matched words in a student answer. The idf of each word is calculated based on the Bag-Of-Word(BOW) generated from the word-answer matrix. This is a good measure to reflect whether prominent keywords in correct answers show up in the student answer.
- *cosine similarity* is calculated using tf-idf vectors of the student answer and the referred correct answers.
- *weighted text similarity*: Wu & Palmer similarity is a knowledge-based measure for text similarity [18], which is based on word similarities. More specifically, we formalize the text similarity between the student answer **s** and the correct answers **c** as sentences. We construct a domain specific word list d for the specific domain by assigning higher weight to domain specific words. Then the text similarity is calculated by weighting the similarities of general words $sim_w(s, c)$ and those of domain specific words $sim_d(s, c)$.
- *Latent Semantic Analysis* (LSA, Landauer and Dumais, 1997): is a computational method which aims to represent a corpora of natural text using the latent subspace. This subspace reflects the weight of each word in each answer so that similar correct answers share similar weight vector of words.

3.1.2 Question (Ques) Model

In domains such as math and science, it is commonly assumed that the relevant knowledge is structured as a set of independent but co-occurring Knowledge Components (KCs). A KC is "a generalization of everyday terms like concept, principle, fact, or skill, and cognitive science terms like schema, production rule, misconception, or facet" [21].

In many Intelligent Tutoring Systems (ITSs) such as Cordillera, completion of a tutor question requires students to apply multiple KCs. By including KCs in our model, we wish to guide the learning process in distinguishing between different

types of questions. Moreover, utilizing KCs is helpful for exploiting the homogeneity among questions. The central idea of Ques model is to build a *Q-matrix* to represent the relationship between individual questions and KCs. Q-matrices are typically encoded as a binary 2-dimensional matrix with columns representing KCs and rows representing questions. Previous researchers have focused on the task of generating or tuning Q-matrices based upon a dataset [1, 20]. For the present work we employ a static Q-matrix manually generated from domain experts.

Additionally, for each question we also include a feature named *questionDifficulty*. It has consistently been selected as one of the important features in our previous work on exploring various state features for modeling student learning [9]. *questionDifficulty* is defined as difficulty level of a question and its value is roughly estimated from the training corpus based on the percentage of answers that were correct on the question in the training dataset.

3.1.3 Student (Stu) Model

Student modeling is an important component for any interactive e-learning environment so that the system can adapt its behaviors based on student needs and knowledge [3]. There are many techniques for generating student models and among them, Bayesian Knowledge Tracing (BKT) [10] is the most widely used. Fundamentally, the BKT model can be seen as a Hidden Markov Model with two hidden states: learned and unlearned. They are defined based on whether a student has mastered the target knowledge or not. BKT keeps a running assessment of the probability that a student is in the learned state based on the student's past history of performance (e.g. *correct*, *incorrect*). BKT assumes that student learning process is a Markov Chain in that at each time $t+1$, the probability of a student has learned the knowledge p^{t+1} is only dependent on his learning state at time t.

Our Stu model used the outputs of the BKT, that is the probability that a student is in the learned state after answering n questions, denoted as $p(S^n = learned)$ as state features. Moreover, our Stu model is KC-specific in that for each of domain KCs, our model will include one probability of being in the learned state on the corresponding KC in the Stu model. Our goal is to use these KC specific probabilities to predict whether the student will answer the next question correctly. Additionally, we also included student KC-specific pretest scores which measures student initial incoming competence.

Therefore, our final Stu model includes a combination of KC-specific learning probabilities calculated from BKT and the student KC-specific pretest scores.

3.1.4 Composite Feature Space

In this part we will explore state features representing the underlying connections between the Ques and the Stu models. As described above, KCs are involved in both Ques and Stu models and thus we hypothesized that a student's performance on a problem should depend on the KCs involved in the problem and the student's performance on corresponding KCs. Hence, we conduct the Cartesian product (CP) using the Ques and Stu models. Additionally, we applied the clustering on the Stu model based on their learning states and pretest scores. Compared with the original features in the Stu model, using student clustering can be seen as more compact representation. Here we used Gaussian Mixture Model, which is a type of soft-clustering methods. Similarly, we hypothesized that the students with similar patterns in Stu clusters may have similar performance on certain types of questions and thus we also conduct the Cartesian product using the student clustering features and Ques vector.

3.2 Six Classifiers

Prior research on ASAG successfully explored several classic ML methods which included: Naive Bayes (NB), Logistic Regression (LR), Decision Tree (DT), Artificial Neutral Network (ANN), and Support Vector Machine (SVM). In recent years deep learning model has been widely used in computer vision and image processing. In this paper, we will compare Deep Belief Networks (DBN) [5] against those five classic ML methods. Given the space constraints, we only briefly describe DBN in the following paragraphs.

DBN is one of the most widely implemented deep learning models. Through the unsupervised pre-training in the first stage, DBN is able to extract the latent features that are more representative than the original input features. Given the input features, DBN first utilizes the stacked Restricted Boltzmann Machine (RBM) layers to automatically extract the high-level features. After the feature extraction in pre-training phase, the weights in these layers are then folded into neural networks for supervised training. Since the capacity of feature extraction mainly lies in the pre-training phase, we now present the mechanism of RBM.

RBM is a restricted version of Markov Random Field. It consists of two layers of variables, visible units V and hidden units H. From the perspective of feature extraction, V stands for the original feature inputs and H denotes the extracted feature representation. The joint distribution of V and H is defined by an energy-based probabilistic model, as follows:

$$P(V,H) = \frac{exp(-E(V,H))}{Z}, \quad Z = \sum_{V,H} exp(-E(V,H)) \tag{1}$$

where the energy function $E(V,H)$ is defined to be:

$$E(V,H) = -V^T W H - B^T V - C^T H. \tag{2}$$

In the above equation, W denotes the weights between V and H. Specifically, $W_{i,j}$ represents the weight between V_i and H_j, and B, C denote the biases for visible units and hidden units, respectively. The denominator Z serves as the normalizer for the probability distribution.

Given that each unit of V or H is independent with other units in the same layer, the conditional distribution is fully factorial and can be easily derived. Due to the intractability of gradient computation brought by the factor Z, the training of RBM (i.e., pre-training phase) follows the Contrastive Divergence algorithm [14], which executes K steps of alternating Gibbs sampling to approximate the gradient. The details can be found in [4].

4. DATA DESCRIPTION

Our training corpus was collected from Cordillera [8, 21], a Natural Language ITS that teaches students introductory college physics. The domain consists of a subset of the physics work-energy domain, which is characterized by eight primary KCs including Kinetic Energy, Gravitational Potential Energy, Spring Potential Energy, and so on. In Cordillera, students interact with tutor by means of natural language entries, and currently the Natural Language understanding module in Cordillera is using human interpreters referred as the language understanding wizard [6]. The *only* task performed by the human wizards is to match student answers to the closest response from a list of potential correct or incorrect responses.

Our training corpus involves 158 students. The data collection consists of the following stages: 1) background survey; 2) studying textbook and prerequisite materials, 3) taking a pretest; 3) training on Cordillera, 4) and taking a post test. In total there are 482 different questions involved in the training corpus and it takes students roughly 4-9 hours to complete the training. Our training corpus includes sequences of tutorial dialogue interactions between students and Cordillera, one sequence per student, and the average number of Cordillera-student interactions is more than 280 per student. For each interaction in a sequence, it consists of a tutor question, a student answer to the question, and two output labels ***correct*** or ***incorrect*** based on human wizards inputs. Thus, *these human manually generated binary labels function as ground truth in our training corpus.*

Based on the definition in [7], our training corpus included **16228** short answers selected from a total of 27868 dialogues. The average length of student answers in our corpus is **7.6** words. **61.66%** of training corpus is labeled as "correct" while the rest are labeled as "incorrect". A series of standard natural language pre-processings including stop word removal, tokenization, punctuation removal and word correction, have been conducted on our training corpus. Additionally, we also conducted domain-specific pre-processing, which includes expanding acronyms to their full forms and removing quantitative questions with equations.

5. EXPERIMENTS

To evaluate the effectiveness of various features from Ans, Ques, and Stu models individually, combined, and/or composite features generated from these three models, we use two ubiquitously implemented classifiers - LR and SVM in Experiment 1. Then in Experiment 2, we will compare DBN against five classic ML classifiers on the best feature model produced in Experiment 1.

5.1 Experiment 1: Exploring Feature Space

For Ans model, we use the five Ans features described in 3.1.1. For Ques model, we include 9 Ques features (one is *questionDifficulty* and the other eight are KC-based Q-matrix features, one feature per KC) and for Stu model, we include 16 Stu features (8 KC-based learning parameters and 8 KC-based pretest scores). Generally speaking, our Experiment 1 can be divided into three stages:

In stage 1, we compare the Ans, Ques, and Stu model individually. Our goal is to investigate whether either Ques or Stu model will be more effective than Ans model for ASAG. In stage 2, we will compare different ways of combining the three basic models. Our results from stage 1 show that Ans-based model alone performs better than either Ques or Stu model (depicted in Section 6.1.1) and thus we mainly explore whether to include the Ques and/or Stu models to the Ans-based model in stage 2. Finally, in stage 3, we will compare different ways of generating new features from the three models (depicted in Section 3.1.4) together with the best model learned from stage 2, which is AQS. Table 2 summarize the types of feature models we explored in each stage.

Table 2: Feature Representations.

Feature	Abbr.	Construction
Stage 1 Basic	*A(ns)*	Ans Model
	S(tu)	Stu Model
	Q(ues)	Ques Model
Stage 2 Combined	*AS*	A + S
	AQ	A + Q
	AQS	A + Q + S
Stage 3 Composite	*CF1*	*AQS* + SC (Student Clustering)
	CF2	*AQS* + SC + CP(Q,S)
	CF3	*AQS* + SC + CP(Q,SC)

⋆ CP denotes Cartesian Product.

To quantitatively evaluate the effectiveness of different feature models, we train LR and SVM with 10-fold cross-validation (CV). LR is widely adopted as the prediction model in industry for its efficiency and robustness. On the other hand, SVM is one of the most popular classifier due to its effectiveness and the capability to incorporate different kernels. Here we adopt RBF kernel for our SVM models.

5.2 Experiment 2: Six Classifiers

In Experiment 2, we evaluate six classifiers with 10-fold cross-validation using the best feature model from Experiment 1, CF3. The six classifiers are NB, LR, DT, ANN, SVM and DBN. As for the DBN, we build three hidden layers, with 74, 34, 10 hidden units respectively and the learning rate is set to be 0.01.

Among the six classifiers, NB assumes the state features are conditionally independent given the output label while the other models do not have such strong assumption and thus are able to combine multiple features to make predictions. Since there exist latent connections among our extracted features, we expect that NB would perform poorly compared to other models. While all five remaining classifiers can make use of combined features to explore latent connections among features, their approaches are different: LR only linearly combines features; DT synthesizes the features at different branches to make predictions; the hidden layers in ANN and the kernel function of SVM can effectively achieve the non-linear feature mapping; while SVM and ANN utilize the relatively fixed pattern for feature combination, DBN enables the extraction of more representative features via a separate unsupervised pre-training procedure. Although the best model CF3 already contains composite features, we expect the DBN can further leverage the latent connections among features that cannot be manually captured in CF3.

6. RESULTS

Five widely used measures, Accuracy, Area Under the Curve (AUC), Precision, Recall and F-measure are used to evaluate how well various classifiers performed. For precision, recall and F-measure, we treat incorrect answers as the positive class because it is more important for the system to know when the student answer is incorrect.

6.1 Experiment 1: Exploring Feature Space

In the following, we will report our results from each stage listed in Table 2. Given that *A(ns)* (Ans model) is the fundamental model studied in previous research, it will be our baseline model for comparisons across three stages.

6.1.1 Stage 1: Three Basic Models

We first compare Ans, Ques and Stu model separately and Table 3 shows the 10-fold cross-validation results. In Table 3, the best performance of corresponding classifier with respect to each measure is in bold and the best value of each measure is marked *.

Table 3: Performance of Basic Models.

Classifier	Evaluation	*A*	*S*	*Q*
LR	Accuracy	**0.646**	0.616	0.633
	AUC	**0.589**	0.499	0.548
	Precision	**0.564**	0.025	0.425
	Recall	0.342	0.001	**0.548**
	F-measure	0.426	0.002	**0.478**
SVM	Accuracy	**0.728**∗	0.540	0.636
	AUC	**0.654**∗	0.546	0.567
	Precision	**0.830**∗	0.422	0.551
	Recall	0.331	**0.572**∗	0.271
	F-measure	0.474	**0.486**∗	0.364

⋆ The majority class is 0.617.
⋆ '∗' is for the highest value of each measure across all models.

Table 3 shows that all three models beat the majority class baseline (0.617) except for the case of applying SVM on Stu model. As expected, when using either LR or SVM, Ans model outperforms Stu and Ques models on Accuracy, AUC and precision. For the other two measures, Stu model provides the best Recall and F-measure when using SVM and Ques model yields the best Recall and F-measure when using LR. Moreover, when comparing LR and SVM, Table 3 shows that SVM classifier seems to be more effective than LR in that the highest values of five measures are all generated by SVM, marked *. More specifically, for Ans model, SVM outperforms LR on all the measures except Recall; for Stu model, SVM outperforms LR on every measure except for Accuracy; finally, for Ques model, SVM outperforms LR on three out of five measures, the exceptions are recall and F-measure.

Overall, while the Ans model generate the best Accuracy, AUC and Precision, the best Recall and F-measure are generated using either the Ques model for LR or the Stu model for SVM. Therefore, we expect combining the Ques and Stu model with Ans model would result in more effective models.

6.1.2 Stage 2: Three Combined Models

To test the effectiveness of combining multiple features, we show the 10-fold CV performance of A, AQ, AS and AQS by applying LR and SVM respectively in Table 4.

Table 4: Performance of Combined Features.

Classifier	Evaluation	*A*	*AQ*	*AS*	*AQS*
LR	Accuracy	0.646	0.719	0.712	**0.768**
	AUC	0.589	0.696	0.690	**0.753**
	Precision	0.564	0.656	0.663	**0.737**
	Recall	0.342	0.591	0.576	**0.671**∗
	F-measure	0.426	0.621	0.616	**0.703**
SVM	Accuracy	0.728	0.784	0.777	**0.822**∗
	AUC	0.654	0.731	0.733	**0.781**∗
	Precision	0.830	0.880	**0.881**∗	0.876
	Recall	0.331	0.505	0.513	**0.615**
	F-measure	0.474	0.641	0.649	**0.723**∗

Table 5: Performance of Composite Features.

Classifier	Evaluation	*A*	*CF1*	*CF2*	*CF3*
LR	Accuracy	0.646	0.786	0.802	**0.810**
	AUC	0.589	0.769	0.784	**0.794**
	Precision	0.564	0.736	0.764	**0.774**
	Recall	0.342	0.692	0.707	**0.720**
	F-measure	0.426	0.713	0.734	**0.746**
SVM	Accuracy	0.728	0.835	0.830	**0.848**∗
	AUC	0.654	0.799	0.824	**0.850**∗
	Precision	0.830	**0.887**∗	0.778	0.769
	Recall	0.331	0.649	0.795	**0.859**∗
	F-measure	0.473	0.750	0.787	**0.811**∗

⋆ $CF1$ AQS + Student Clustering (SC).
⋆ $CF2$ AQS + SC + Cartesian product(Ques, Stu).
⋆ $CF3$ AQS + SC + Cartesian product(Ques, SC).

It is observed that by adding either Ques or Stu model into Ans model, the effectiveness of resulted models is greatly improved on each of five measures. For example, the Accuracy increases from 0.646 for Ans model to 0.719 for AQ model, and 0.712 for AS model under LR. We can observe the same pattern when SVM is applied. For both LR and SVM classifier, it seems that AQ and AS have comparable performance.

AQS, the combination of all three models, outperforms either AQ or AS for both LR and SVM on all five measures except on Precision by SVM where AS has a slightly higher value (0.881) than AQS (0.876). Therefore, it suggests that Stu and Ques model indeed contribute different information to ASAG task. Similarly, across three models, Table 4 shows that the SVM classifier seems to be more effective than LR in that the best of each of the five measures (those marked *) are generated by SVM except for Recall where the best value 0.671 is generated by LR on AQS model.

6.1.3 Stage 3: Three Composite Models

Given that AQS performs as the best model in Stage 2, we explore whether the effectiveness of classifiers can be further improved by adding composite features. Table 5 shows the performance of CF1, CF2 and CF3.

Tables 4 and 5 show that CF1 is more effective than AQS on every measure when using SVM and on four out of five measures except on Precision using LR. It suggests that the using student clustering can indeed further improve the performance of either LR and SVM.

The improvement from CF1 to CF2 and CF3 mainly stems from the power of Cartesian product. Furthermore, the difference between CF2 and CF3 lies in the different choices of features used for Cartesian product. The result shows that there exists stronger association between the latent student clusters and Ques model than that between Stu model and Ques model. Overall, SVM outperforms LR throughout CF1

to CF3 in that the best of five measures (those marked *) are all generated by SVM in Table 5.

To summarize, the performance of SVM dominates LR when using individual feature models, combined models, and composite models. With only one exception, the best of each of the five measures (those marked *) are all generated by SVM across all three stages. Finally across the nine models, the best model for both LR and SVM is CF3 in that CF3 is more effective than the other eight models on every measures using LR and on four out five measures except on Precision using SVM. Therefore, CF3 is selected for Experiment 2.

6.2 Experiment 2: Six Classifiers

Table 6 shows the performance of the six ML classifiers on CF3: $AQS + \text{SC} + \text{CP(Q,SC)}$ using 10-fold cross-validation. From the results, we draw the first conclusion that NB falls behind other classifiers with a large margin of 18% except on Recall. As expected, LR, DT, ANN, SVM and DBN outperform NB in all the evaluations due to the capacity of combining features and NB's strong independent assumption. Table 6 shows that DBN yields the highest Accuracy, AUC, Precision and F-measure while SVM reaches the best recall value of 0.859 closely followed by DBN. For AUC and F-measure, we have the values in the increasing order for NB, LR, DT, ANN, SVM, and DBN. Overall, our results suggest that DBN performs the best among the six classifiers followed by SVM and NB performs the worst.

Table 6: Comparing the Six Classifiers

Evaluation	NB	LR	DT	ANN	SVM	DBN
Accuracy	0.631	0.810	0.825	0.837	0.848	**0.850***
AUC	0.667	0.794	0.813	0.827	0.850	**0.890***
Precision	0.511	0.774	0.775	0.791	0.769	**0.830***
Recall	0.823	0.720	0.765	0.784	**0.859***	0.838
F-measure	0.631	0.746	0.770	0.787	0.811	**0.834***

7. CONCLUSION

In this paper we tackled the task of ASAG through feature engineering and exploration of better ML approaches such as DBN. For feature engineering, we utilized two other models: Ques and Stu models and explored various combined and composite feature representation. Our results showed that by utilizing the composite features, we obtain an AUC improvement of around 35% and 30% and F-measure improvement of around 75% and 72% on LR and SVM respectively as compared with using Answer-based features only. The comparisons among different classification models shows that DBN outperforms all other methods on Accuracy, AUC, Precision and F-measure. On Recall, DBN performs slightly worse than SVM. Furthermore, the experiment has led to some interesting observations: (1) The clustering of student, as a more compact representation, leads to more discriminative features when combined with question features using Cartesian product. (2) While SVM results in better Accuracy, the composite feature representation brings less improvement on SVM than LR probably because we used RBF kernel in our SVM models which allows the classifier to operate in an infinite-dimension of feature space.

8. ACKNOWLEDGMENTS

This research was supported by the NSF Grant 1432156 "Educational Data Mining for Individualized Instruction in STEM Learning Environments".

9. REFERENCES

[1] T. Barnes. The q-matrix method: Mining student response data for knowledge. 2005.

[2] S. Basu, C. Jacobs, and L. Vanderwende. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, 2013.

[3] J. E. Beck and B. P. Woolf. Using a learning agent with a student model.

[4] Y. Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2009.

[5] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 2007.

[6] N. O. Bernsen and L. Dybkjaer. *Designing Interactive Speech Systems: From First Ideas to User Testing*. Springer-Verlag New York, Inc., 1997.

[7] S. Burrows, I. Gurevych, and B. Stein. The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, 2015.

[8] R. Carolyn. Tools for authoring a dialogue agent that participates in learning studies. *Artificial Intelligence in Education: Building Technology Rich Learning Contexts that Work*, 158:43, 2007.

[9] M. Chi, K. VanLehn, D. Litman, and P. Jordan. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction*, 2011.

[10] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 1994.

[11] S. K. D'Mello, B. Lehman, and A. Graesser. A motivationally supportive affect-sensitive autotutor. In *New perspectives on affect and learning technologies*. Springer, 2011.

[12] M. O. Dzikovska, A. Isard, P. Bell, J. D. Moore, N. Steinhauser, and G. Campbell. Beetle ii: an adaptable tutorial dialogue system. In *Proceedings of the SIGDIAL 2011 Conference*, pages 338–340. Association for Computational Linguistics, 2011.

[13] D. Higgins, J. Burstein, D. Marcu, and C. Gentile. Evaluating multiple aspects of coherence in student essays. In *HLT-NAACL*, 2004.

[14] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 2002.

[15] G. B. Huang, H. Lee, and E. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *CVPR*, 2012.

[16] C. Leacock and M. Chodorow. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 2003.

[17] A.-r. Mohamed, G. Dahl, and G. Hinton. Deep belief networks for phone recognition. In *NIPs*, 2009.

[18] M. Mohler and R. Mihalcea. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th EACL*, 2009.

[19] S. G. Pulman and J. Z. Sukkarieh. Automatic short answer marking. In *Proceedings of the second workshop on Building Educational Applications Using NLP*.

[20] K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement.*, 1983.

[21] K. VanLehn, P. W. Jordan, and D. J. Litman. Developing pedagogically effective tutorial dialogue tactics: experiments and a testbed. In *SLaTE*. Citeseer, 2007.

Posters and Demo

Redefining "What" in Analyses of Who Does What in MOOCs

Alok Baikadi[1,2], Christian D. Schunn[1], Yanjin Long[1,2], Carrie Demmans Epp[1,2]
[1]Learning Research and Development Center
[2]Center for Instructional Development and Distance Education
University of Pittsburgh
{baikadi, schunn, ylong, cdemmans}@pitt.edu

ABSTRACT

To advance our understanding of learning in massive open online courses (MOOCs), we need to understand how learners interact with course resources. Prior explorations of learner interactions with MOOC materials have often described these interactions through stereotypes, which does not account for the full spectrum of potential learner activities. A focus on stereotypes also limits our ability to explore the reasons behind learner behaviors. To overcome these shortcomings, we apply factor analysis to learner activities within four MOOCs to identify emergent behavior factors. The factors support characterizations of learner behaviors as driven heavily by types of learning activities and secondarily by time/topic; regression revealed demographic factors (especially country and gender) associated with these activity and topic preferences. Both factor and regression analyses revealed structural variability in learner activity patterns across MOOCs. The results call for a reconceptualization of how different learning activities within a MOOC are designed to work together.

Keywords

MOOCs; learning analytics; online learning; factor analysis

1. INTRODUCTION

With the increasing popularity of massive open online courses (MOOCs), the need to investigate the relationships among learner characteristics, learner-selected activities, and learning outcomes has become critical. Determining these relationships can help us understand how people learn within MOOCs and inform MOOC design and pedagogy. Prior work identified different learning-activity patterns [1, 3] and investigated the relationship between certain types of learning activities and outcomes [7]. Many of these studies were conducted in the context of a single domain or MOOC (e.g., [1]). Furthermore, little work has investigated how demographic variability could lead to different behavioral patterns in MOOCs, leaving an open question: Can the identified patterns be generalized across instructional domains and populations?

Until recently, studies of learning within MOOCs focused more on the number of learners being served than pedagogy [6]. This focus on their size has left many facets of MOOCs underexplored and poorly understood [1]. These aspects include a need to understand how learners engage with MOOCs [1], their behavior patterns, and their motivations [3]. Understanding these factors may allow us to design courses that support the learning activities and outcomes that learners want.

We investigate learning patterns in four MOOCs based on learner activities across courses from different disciplines. We used the activity-centered data reduction technique of factor analysis to identify the underlying course activities that describe learner activity patterns within each offering of the selected MOOCs. The factor analyses applied to 10 MOOC offerings enabled us to identify 1) factors that are common to most of these MOOCs and 2) factors that are less common. Regression analyses were then used to examine the relationship between learner demographic variables and their participation on each factor. These analyses support the distinctions between factors and the presence of varied factors across MOOCs.

This investigation is among the first to identify and compare activity patterns and demographic influences across learning domains. The results improve our understanding of learner behaviors across contexts and could inform the design of more domain-sensitive learning experiences.

2. LEARNER ACTIVITIES IN MOOCS

Research into MOOCs has spanned a range of topics, with recent discussions becoming more nuanced. Work that has investigated how learners interact with a MOOC [5] found that their behaviors can be characterized through a set of trajectories rather than the commonly used completion and attrition model. These trajectories through graded assignments and lecture videos within computer science MOOCs characterize how different types of learners used some of the course materials to support their learning activities [1]. The identified usage patterns included those who mostly watched lectures, mostly submitted assignments, performed some combination of these activities, downloaded course resources, or registered but did very little.

Some researchers have taken the next step by linking these types of activities (watching video lectures, submitting assignments, and discussion forum activity, types of questions asked) to course performance (certificate earned, learning outcomes and gains, course completion) [2, 7]. To obtain a better understanding of how these and other factors influence learner success within MOOCs, the relationships among socio-demographic variables, student activities, and learner success have been explored. The most common predictors of certificate earning and completion were prior education [2], sex [4], and country of origin [4].

3. MOOC CORPUS

Data from the 132,324 learners who performed at least one action (taking a quiz, posting to the forum, or watching a video) in 4 of the University of Pittsburgh's Coursera MOOCs were used. To describe learner activities within a range of course types and explore generalizability across disciplines, courses from different domains were chosen: health sciences (nutrition for health and clinical terminology), education (accountable talk), and public health (disaster preparedness). Data from multiple offerings (Jan. 2013 – Dec. 2015) of the same course were used when available.

The courses lasted 6 or 7 weeks. The core materials for each week consisted of video lectures and a quiz. Some weeks included assignments, disaster preparedness used peer-assessment, and accountable talk had a project. Clinical terminology incorporated multimedia modules that enabled the learner to interact with learning resources. Since these modules presented core content, they were labeled as lectures. Only the Clinical Terminology instructors explicitly encouraged discussion forum use and provided study tips. This variability provided a cross-section of course formats that enables us to identify learner activities that apply across courses and that are specific to a course. We used the activity counts for each forum, quiz, and lecture video.

4. RESULTS

4.1 Learner Activities

Factor analysis with varimax rotation was used to reduce the dimensionality of the data and identify learners' underlying behavioral tendencies. Course activities that at least 1% of active learners performed were used. To test the stability of the patterns, a separate factor analysis was conducted for each course offering. Factors that accounted for at least 5% of the variance were kept.

In 3 of the 4 courses, activities were largely grouped into 4 factors: lecture activity, quiz activity, forum participation and participation in activities from weeks 1 and 2. In contrast, clinical terminology shows more depth in weekly content: lecture activity is represented by 4 factors, each capturing a 1-2 week span. For quizzes, we see three factors: summative quizzes presented at the end of each module, early quiz activities, and later quiz activities.

4.2 Predicting Activities Using Demographics

We calculated a factor score for each learner, which indicates a tendency towards the behavior described by that factor. For example, a learner with a high score for the lectures factor would have viewed more lectures than one with a low score. A general linear model (GLM) was used to predict learner factor scores from learners' socio-demographic characteristics. Only those (n = 2963) with individual demographic profiles were included. We applied GLM to courses that had contrastive factor structures: the second offering of nutrition for health represented those with media-based factors and the first offering of clinical terminology represented those with time-based factors.

For clinical terminology, we aggregated early lecture factors, late lecture factors, and quiz factors to create factors that were comparable to the other courses. We then ran a generalized linear model predicting each of these aggregated factors.

Each factor is influenced differently by learner demographics and are contrasted between the two courses. For example, the early lecture watching factor from nutrition for health was more strongly associated with female learners than males. This was not the case for clinical terminology. Late lecture watching activity was predicted by learner age for both courses. However, a difference in factor scores for the younger and older populations for those in the middle age groupings is visible between the courses. Within clinical terminology, we also see that some age groups are more active earlier in the course than later. Additional differences in how demographic variables predict factors are visible when considering learners' quiz participation and their continent of residence. Similar factor scores are seen for those who live in Asia and North America when considering learner activities within clinical terminology. This similarity does not hold across courses; learners from Asia and Europe appear to be more similar in their quiz taking habits when considering the data from nutrition for health.

5. CONCLUSION

Our factor and regression analyses across multiple offerings of the same course show that learner behaviors are relatively consistent across time. However, differences in factors across courses suggest that design and domain affect how learners select learning content and activities, which requires further study.

Our work is among the first applications of exploratory factor analyses across learner activities within MOOCs from different domains. Prior work has focused on a person-based approach that describes the behavior patterns of individuals by assigning them to canonical groups. This work, therefore, provides a new lens to examine the full range of learner behaviors in MOOCs.

6. REFERENCES

[1] Anderson, A., Huttenlocher, D., Kleinberg, J., and Leskovec, J. 2014. Engaging with Massive Online Courses. In *Proceedings of the 23rd International Conference on World Wide Web*, ACM, 687–698. DOI = http://doi.org/10.1145/2566486.2568042

[2] Breslow, L. B., Pritchard, D.E., DeBoer, J., Stump, G.S., Ho, A.D., and Seaton, D. T. 2013. Studying learning in the worldwide classroom: Research into edX's first MOOC. *Research & Practice in Assessment* 8: 13–25.

[3] Gasevic, D., Kovanovic, V., Joksimovic, S., and Siemens, G. 2014. Where is research on massive open online courses headed? A data analysis of the MOOC Research Initiative. *The International Review of Research in Open and Distributed Learning* 15, 5.

[4] Kizilcec, R.F. and Halawa, S. 2015. Attrition and Achievement Gaps in Online Learning. In *Proceedings of the 2nd ACM Conference on Learning @ Scale*, ACM, 57–66. DOI = http://doi.org/10.1145/2724660.2724680

[5] Kizilcec, R.F., Piech, C., and Schneider, E. 2013. Deconstructing Disengagement: Analyzing Learner Subpopulations in Massive Open Online Courses. In *Proceedings of the 3rd International Conference on Learning Analytics and Knowledge (LAK)*, ACM, 170–179. DOI = http://doi.org/10.1145/2460296.2460330

[6] Kovanović, V., Gašević, D., Joksimović, S., Siemens, G., and Hatala, M. 2015. MOOCs in the News: A European Perspective. In *Proceedings of "WOW! Europe embraces MOOCs."*

[7] Wang, X., Yang, D., Wen, M., Koedinger, K., Rosé, C. P. (2015). Investigating how student's cognitive behavior in MOOC discussion forums affect learning gains. In *Proceedings of the 8th International Conference on Educational Data Mining (EDM)*.

Text Classification of Student Self-Explanations in College Physics Questions

Sameer Bhatnagar
Polytechnique Montreal

Michel Desmarais
Polytechnique Montreal

Nathaniel Lasry
John Abbott College

Elizabeth S. Charles
Dawson College

ABSTRACT

This study looks at the text data generated from the Asynchronous Peer Instruction tool, DALITE. The goals of this work are two-fold: i) to determine whether the words students use in their self-explanations can be predictive of their success on the related multiple-choice item, or even reveal their uncertainty about the concept being tested; and, ii) to determine if the collection of words used by a student over the course of a semester using DALITE can predict their end-of-semester learning outcomes. Through the course of this study, we examine the effectiveness of different statistical models and document representations to explain these data. Weak results suggest richer syntactic ans semantic models of text are needed.

1. INTRODUCTION

The Distributed Active Learning Integrated Technology Environment (DALITE)[2], implements an original peer instruction paradigm that relies on students providing a rationale to their choice over multiple-choice questions (MCQ). After every MCQ, the student is prompted to provide the rationale for their choice. Once provided, the student is shown a few other students' rationales for the same choice, and for an alternate choice. If the answer was right, the alternate choice shown is for a wrong answer, else it is the right answer's rationales. The student can then decide to change their choice or not. This instruction paradigm has recently been integrated into the EdX platform and we believe it has a great future in MOOCs and other environments where educational crowdsourcing bootstraps instructional content. However, for the bootstrap to be effective, a good understanding of the process of learning from this type of content is crucial. This paper reports on early analysis of student rationales with this aim in mind, using a text classification framework. For this particular study, we are interested in

- identifying students who are unsure about their answers (as revealed by when they switch from right-to-wrong, or wrong-to-right in DALITE). Are there linguistic patterns for students who are uncertain?
- studying the effect of the teacher on the development of their students' language. Is there a teacher effect?
- documenting group differences in language use, for subpopulations such as strong vs. at risk students, or male vs. female. [6] discusses the gender gap in performance in college physics classrooms. This was observed in a previous study of ours looking at DALITE as well[1]. Is there a measurable difference between the language used by strong students and weak ones? Are there gender differences?
- finding minimally disruptive, low-stakes, language based predictors of student failure, as early in the semester as possible. Can the results of DALITE questions assigned prior to any of the three midterms predict which students ultimately fail?
- which classification algorithms perform the best in this context? What document representations optimize classifier performance for the different target variables?

2. DATA AND METHODS

2.1 Corpus Statistics

The dataset is made up of student-generated self-explanations for 80 different DALITE items (conceptual physics questions). On average, 97 students attempted each item, writing explanations for each question with an approximate length of 32 words, with a type-token ratio of 0.87. The average number of unique words used by all students to answer any given one item was 310. The 140 students in this study came from three different colleges in the province of Quebec, Canada. The course material was surrounding what would normally be freshman physics in the U.S. Besides collecting midterm grades and final course grades, each student also completed the Force Concept Inventory[4], at the beginning of the term, as well at the end. The normalized pre-post gain (or Hake gain) on this questionnaire has become a standard measure in the physics education research community. More aggregate statistics of the dataset rest are more fully described in [1].

2.2 Statistical Models

Significant amount of work was done in comparing different statistical learning algorithms for text classification. One of the simplest yet most effective text classification approaches

is the Naive Bayes classifier[7]. In datasets when vocabulary size was small, [8] compared different event models for the Naive Bayes family of classifiers, finding that the multivariate Bernoulli model (where the components of each document vector are binary, modeling simply the presence or absence of a word), performed better for text classification than its multinomial counterpart (where document vectors are the counts of the different terms in that document). [5] shows that Support Vector Machines (SVM) are well suited to the task of text classification, due to three factors inherent to the nature of the task: high dimensional feature space, many relevant features (dense concept vectors), but sparse document vectors. Finally, we explore the utility of a k-nearest neighbor classifier in this setting as well, based on the intuition that the document vectors might not be linearly separable.

2.3 Document Vector Representations

This study also aims to explore different choices of document representation. The most basic choice would have the elements of document vectors simply containing raw word counts (we ensure that the words in the original questions item text are always included in the term-document matrices).[9] showed that shifting importance to rarer words across a corpus would improve classifier effectiveness. We also look at N-grams to relax the independence assumption between words, but this may require more data than we have to avoid sparsity (we only go up to bigrams). There is an interest in also adding syntactic information, such as part-of-speech (POS) tags, and represent documents as bags of POS-tags (e.g. since there is an important difference in physics between using the word "force" as a verb or as a noun, which could reveal a misconception if students use it incorrectly). Finally, document vectors can also be represented for their semantic content. One of the most successful techniques for this is Latent Semantic Analysis[3], which relies on a truncated singular value decomposition of term co-occurrence matrices. This allows us to approximately represent documents in a lower dimensional space, and typically removes noise such that document vectors that are similar in meaning, cluster together. The sensitive choice in such latent factor models is the choice of how many factors will be kept after the matrix decomposition. We do a grid search over different possible number of dimensions to reduce to, ranging from 2 to 10, and pick the model that performs best in cross-validation.

3. DISCUSSION

None of the results are presented here, due to space limitations.[1].Our research team started this study with the following question: do students in different cognitive states, use different words to explain their thinking when answering conceptual questions? In general, the poor performance of most of the statistical models studied herein tends to confirm the intuition behind the body of work centered around Latent Semantic Analysis: in most cases, the mere occurrences of the words is not enough to discriminate strong students from weak ones, and that such datasets can be too noisy and sparse. The inability of all these models to predict item-level outcomes, such as getting the answer correct, or whether a student is about to switch their answer, leads us to believe that richer syntactical and semantic representations will be required.

[1]All scripts used to get the results, for this study are available at sameerbhatnagar.github.io/

4. FUTURE WORK

The most important facet of DALITE that has not yet been studied lies in the patterns in student preferences: when students are on the page where they can read their peers' rationales, and are asked to reconsider their original answer choice, they are also prompted to *select which, if any, of their peers' rationales they thought was most convincing.* This 'crowdsourcing' of high quality, peer-assesed rationales if very healthy for the future of DALITE, but is also fertile ground for research related to the current study: what distinguishes language that is effective to convincing to students (whether for the right answer, or the wrong one)?

5. ACKNOWLEDGMENTS

We would like to thank the teachers who participated in this study, without whose support this work would not be possible: Chris Whittaker (Dawson College), Kevin Lenton (John Abbott College), and Kevin Lenton (Vanier College). This work was funded through *Programme de Recherche sur l'Apprentissage et l'Enseignement* from the government of Quebec. Finally, we remain indebted to all our students who actively contributed to DALITE through their participation.

6. REFERENCES

[1] S. Bhatnagar, M. Desmarais, C. Whittaker, N. Lasry, M. Dugdale, and E. S. Charles. An analysis of peer-submitted and peer-reviewed answer rationales, in an asynchronous peer instruction based learning environment.

[2] E. Charles-Woods, C. Whittaker, M. Dugdale, N. Lasry, K. Lenton, and S. Bhatnagar. Designing of dalite: Bringing peer instruction on-line. In N. Rummel, M. Kapur, M. Nathan, and S. Puntambekar, editors, *Computer Supported Collaborative Learning.*

[3] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[4] D. Hestenes, M. Wells, and G. Swackhamer. Force concept inventory. *The physics teacher*, 30(3):141–158, 1992.

[5] T. Joachims. *Text categorization with support vector machines: Learning with many relevant features.* Springer, 1998.

[6] L. E. Kost, S. J. Pollock, and N. D. Finkelstein. Characterizing the gender gap in introductory physics. *Physical Review Special Topics-Physics Education Research*, 5(1):010101, 2009.

[7] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[8] A. McCallum, K. Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

[9] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

Automated Feedback on Group Processes: An Experience Report

Marcela Borge
Pennsylvania State University
301C Keller Building
University Park, PA 16802
mborge@psu.edu

Carolyn P. Rosé
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
cprose@cs.cmu.edu

ABSTRACT
We report on an effort to evaluate the efficacy of automated assessment and feedback of the quality of collaborative discourse in the context of an online project based course. Results of automated assessment and impact on collaborative process are evaluated over a semester-long course.

Keywords
Collaborative learning, automated process assessment

1. INTRODUCTION
In this paper we report on an effort to evaluate the efficacy of automated assessment and feedback of group processes in the context of an online project based course. It is well known that the positive effects of collaborative learning are not guaranteed. Instead, those benefits depend upon the quality of collaborative interactions that occur during activity [1]. This is problematic since most students lack the cognitive skills necessary to engage in high quality collaborative interactions [3]. Research suggests that developing socio-metacognitive expertise, the ability to understand, monitor, and regulate collective thinking processes that occur during collaboration, can help to mitigate group dysfunction and optimize collaborative interactions [4].

We have been working on developing activity design models to inform the design of Computer Supported Collaborative Learning (CSCL) systems to support socio-metacognitive development [4]. In this paper, we describe an approach to automated, collaborative discourse assessment and a study we ran in a real educational environment. We focus on two areas of inquiry motivated by emerging research. First, (RQ1) How reliably can we automatically assess collaborative discussion quality and (RQ2) does automated assessment impact future performance differently than human generated feedback?

2. METHODS

2.1 Study Context
The study took place during a 16-week, introductory, undergraduate, online course on information sciences and technology. Fourty-one online students participated in the study, each belonging to one of 14 groups. As part of the course, students were required to read a chapter from the textbook or supplementary materials each week. Students were assigned to teams within the first four weeks of the semester. Then, in weeks five, seven, nine, eleven, and fourteen, students participated in a synchronous discussion related to the reading materials. The discussion sessions were held in a collaborative workspace with chat capabilities called CREATE.

2.2 Research Design
Across the five time-points during which students engaged in a collaborative chat activity, we compared the effect of four different feedback conditions on the quality of collaboration at the next time point. After each of the first four discussion tasks, groups were assigned to one of four feedback conditions that determined the type of feedback they received at that time point.

The study was run as a within-subject manipulation. The four conditions included: (1) no feedback, (2) expert feedback, (3) automated feedback, and (4) best practices. Those in condition one received no feedback about the quality of their processes. Those in condition two received feedback from trained research assistant who would analyze their processes using our coding construct. Condition three received feedback based on automated assessment of processes. Condition four was given feedback based on common strengths and weaknesses of collaborative groups [4] and not based on the group's specific processes. All feedback was worded in a consistent manner such that teams would not know what condition they received.

An assessment of group processes was conducted for each discussion based on the transcripts from the chat environment that housed the activity. Team process measures at the first time point were used to identify groups' initial strengths and weaknesses. Thus, the first assessment was treated as a baseline, and each subsequent measurement, controlling for the previous assessment, was treated as a measure of the effectiveness of the form of feedback experienced after the previous discussion.

2.3 Assessment of Collaborative Discourse Quality
After each discussion session, individual students completed an evaluation of the quality information synthesis and knowledge negotiation in their group.

In the assessment rubric, there are three categories of behavior within each of the two core capacities, with each category assessed on a five-item, ordinal scale. The first core capacity, information synthesis, consists of three categories of discourse behavior: verbal participation, developing joint understanding, and joint idea building. Verbal participation examines the amount of turns of speech contributed by each member relative to the team's total turns of speech. Developing joint understanding evaluates the extent to which teams make an effort to ensure that members fully understand the ideas presented by taking time to reword, rephrase, or ask for further clarification of shared information. Joint idea building focuses on the extent to which team members elaborate on another member's contribution in

order to ensure that information introduced by any member is not ignored or accepted, without discussion.

The second core capacity, knowledge negotiation, also consists of three categories of behavior. These categories are contributing alternative ideas, quality of claims, and norms of evaluation. Contributing alternative ideas evaluates the extent to which teams present and discuss alternative perspectives, claims, or suggestions. Quality of claims focuses on evaluating the extent to which teams provide logical, fact-based evidence and rationale. Norms of evaluation focuses on evaluating the extent to which teams adhere to social norms that promote the development of psychological safety.

Twenty percent of the total data was double coded by the research assistant and another trained graduate student to determine inter-rater reliability of the instrument: $r = 0.86$; $p < 0.001$, Kappa = 0.64; $p < 0.001$. Once each item of a core capacity is rated, they are averaged to produce a single Collaborative Discussion Quality score, which is a continuous value between 0 and 5 that we use to track improvement over time in collaborative discussion processes in the analysis below.

2.4 Automated Assessment

A key component of the study is an evaluation of an automated assessment technique. The six scales that comprise the three dimensions of each of the two core competencies in the assessment rubric were automatically predicted based on distributions of automatically predicted process codes. Training data for the macro level regression model for the 6 scales was a corpus of 13 discussions (with a total of 7015 turns) that were hand coded with a process-analysis coding scheme developed as part of this work. We built on a coding scheme developed for a laboratory study [3], but modified it for use in a real-world classroom setting. Each discussion was hand coded at the turn level using the process analysis and then assessed along the 6 different dimensions. We established inter-rater reliability for this schema of Kappa= .74, indicating substantial reliability.

The automated process analysis models were trained using the LightSIDE tool bench. We extracted a feature space consisting of unigrams, bigrams, POS bigrams, and a line length feature, and used a Logistic regression classifier with L2 regularization to avoid over-fitting. In a leave-one-team-out cross-validation, we achieved an accuracy of 86% and kappa of .77. The assessment needed in order to generate feedback for the study is at the level of the six scales that rate two core competencies, with three dimensions each. We used the counts of predicted process codes per team to predict these six scales using a separate linear function trained using a simple linear regression for each scale.

We expected a drop in performance when applying a model trained in a previous experiment. In the initial week of the study, we used the model trained on the earlier data to generate the six scores per team. In subsequent weeks of the study, we retrained the simple linear regression models to predict hand coded assessment scores from data collected in the current study during the earlier weeks of the semester. The process coding that created the predictor variables for those regression equations was computed using the original trained process coding models.

3. RESULTS

At each of four time points in the course, we collected automated assessments of collaborative process in terms of the six assessment dimensions. Each time, each of three to four groups was assigned a rating on a 5-point scale for each of the six dimensions. The same assessments were also made by human raters in order to assess the quality of the automated rating. Over time, we continued to use the original turn level process models but adapted the simple linear regressions to compute the six scale measures from the counts of the turn level codes using the hand rated data collected in the second course so far. We evaluate the quality of the automated rating by computing a kappa with linear weighting between the sets of automated ratings and human ratings. At time point one, before any data from the second instance of the course was available, the automated ratings were assessed to be at random. By time point two, the weighted kappa was .19. It was better at time point three, specifically .4. And finally, at time point four, it was up to .58. Altogether ratings for 10 sessions of the second course were needed to adapt the models and achieve a weighted kappa of .58.

Given that the automated feedback generated at early time points in the course was based on poor quality assessments, an important question is how much of a negative impact these errors cause for students. We measured the effect of the experimental manipulation using a repeated measures ANCOVA for each scale assessment separately. In each case, the dependent measure was the scale assessment at a time point rated by an expert rater, the covariate being that scale assessment at the previous time point, the independent variable being the condition that generated the feedback received by the team at the previous time point, and time point as a nominal control variable. We did not observe any consistent improvement over time or significant effect of condition on any one of the six scale assessments.

4. CONCLUSIONS

In this paper we addressed important questions related to the automated assessment of collaborative discourse quality in real educational settings. Though the automated process analysis was evaluated as very reliable within the course that provided the training data, the automated assessments in the second run of the course were initially very poor and only improved after 3 weeks of data were collected to use for adapting the prediction models.

5. ACKNOWLEDGMENTS

This work was funded by NSF grant IIS-1320064.

6. REFERENCES

[1] Barron, B. (2003). When smart groups fail. *Journal of the Learning Sciences*, 12(3), 307-359.

[2] Biber, D. & Conrad, S. (2011). *Register, Genre, and Style*. Cambridge University Press.

[3] Borge, M., & Carroll, J. M. (2014). Verbal Equity, Cognitive Specialization, and Performance. In *Proceedings of the 18th International Conference on Supporting Group Work*, 215–225.

[4] Borge, M., Ong Shiou, Y., & Rosé, C. 2015. Design models to Support the Development of High Quality Collaborative Reasoning in Online Settings. In *the Proceedings of the International Conference of Computer Supported Collaborative Learning (CSCL) 2015,* Volume 2, 427-434.

Mining Sequences of Gameplay for Embedded Assessment in Collaborative Learning

Philip Buffum
North Carolina State University
Computer Science
psbuffum@ncsu.edu

Megan Frankosky
North Carolina State University
Psychology
rmhardy@ncsu.edu

Kristy Elizabeth Boyer
University of Florida
Computer & Information Science & Engineering
keboyer@ufl.edu

Eric Wiebe
North Carolina State University
STEM Education
wiebe@ncsu.edu

Bradford Mott
North Carolina State University
Computer Science
bwmott@ncsu.edu

James Lester
North Carolina State University
Computer Science
lester@ncsu.edu

ABSTRACT
This poster presents a sequence mining analysis of collaborative game-based learning for middle school computer science. Using pre-post test results, dyads were categorized into three groups based on learning gains. We then built first-order Markov models for the gameplay sequences. The models perform well for embedded assessment, classifying gameplay sequences with 95% accuracy according to whether the group learned the target concepts or not. These results lay the groundwork for accurate embedded assessment of dyads in game-based learning.

Keywords
Embedded assessment; game-based learning; collaboration; Markov models

1. INTRODUCTION
There is growing recognition of the importance of collaborative learning, in which students work together to solve problems [2, 3]. Collaboration, furthermore, can have an especially beneficial impact in game-based learning, where it has been shown to promote significant student learning gains [4] and provide significant motivational benefits [8], as well as deliver more equitable gaming experiences for diverse learners [1, 6].

Yet collaborative learning presents unique challenges to educational data mining research. While much current work in this field relies on mapping individual students' outputs, student collaboration produces learning that plays out as a joint activity, necessitating different approaches to understanding the underlying processes [7]. Recent work in educational data mining has demonstrated some success in predicting student outcomes in paired learning, as long as both students in the pair have similar initial knowledge [5].

This poster examines collaborative game-based learning in the context of the ENGAGE game-based learning environment, with which middle school students learn about computer science through an overarching narrative situated within a fictional underwater research station. In this study, students played ENGAGE in pairs at a single computer, taking turns with one set of game controls. These two students' inputs were therefore captured within a single gameplay log. The analysis presented here investigates a variation on the traditional learning question of, "Did student *S* learn the concept?" and instead asks, "Did the collaborative partnership *P* result in learning?" By building first-order Markov models on dyads' gameplay logs, we discovered that the gameplay sequences of dyads in which some learning occurred (i.e. at least one of the students learned the material) differed significantly from those in which no learning occurred, and moreover, that we can classify with very high accuracy the learning that occurred on a targeted learning objective.

2. COLLABORATIVE LEARNING TASK
This study focuses on a subset of the ENGAGE game. In ENGAGE'S Digital World level, students learn how computers process data using the binary number system. The current analysis focuses on one room in the game world, in which students integrate the two concepts of *variables* and *binary numbers*, having earlier explored both these individual concepts in isolation from one another. 124 middle school students played the game in pairs; as there is one gameplay trace for each dyad, this produced 62 gameplay traces. We administered individual pre- and post-tests to each student so that we could characterize each student's learning outcomes. The goal of the present analysis is to utilize gameplay logs to predict learning, specifically to investigate how the gameplay of those dyads who scored higher on learning assessments differs from the gameplay of those who did not score higher. Accordingly, having assigned each *individual* student a grade based on pre and post test scores, we then classified student *pairs* into one of three categories: *Learner* (19 dyads), *Prior Mastery* (23 dyads), and *Non-Learners* (20 dyads).

3. RESULTS
The modeling approach aims to identify differences in gameplay sequences between students in the *Learner*, *Prior Mastery*, and *Non-Learner* groups. We began with one of the simplest sequential models of all, first-order observable Markov models. It was expected that more sophisticated models, such as hidden Markov models or Conditional Random Fields, may be needed to characterize the gameplay sequences well; however, as this poster demonstrates, the simplest model was able to classify the gameplay sequences of *Learner*, *Prior Mastery*, and *Non-Learner* groups with high accuracy.

We built separate models for each group (*Learner, Prior Mastery, Non-Learner*) and then determined whether there were significant differences in the models for each group by comparing model fit (in terms of log-likelihood, since the probabilities themselves are very small in magnitude). We performed this pairwise comparison for all three groups, as described below:

1. For each gameplay trace sequence s_i in the Learner group:
   i. Compute $logProb(s_i \mid L_{leave\text{-}i\text{-}out})$ of observing s_i under the Learner model L (trained in a leave-one-out fashion where s_i was the left-out sequence).
   ii. Compute the log-likelihood $logProb(s_i \mid PM)$ of observing s_i under the Prior Mastery model PM trained on all Prior Mastery gameplay sequences.
   iii. Compute the log-likelihood $logProb(s_i \mid NL)$ of observing s_i under the Non-Learner model NL trained on all Non-Learner gameplay sequences.
2. Repeat the analogous process for each gameplay sequence in the Prior Mastery and Non-Learner groups.
3. For each group's sequences, test whether the set of log-likelihoods for that group under its own model is significantly higher than the log-likelihoods for that group under the other groups' models.

The models were significantly different across *Learners*, *Prior Mastery*, and *Non-Learner* groups, as shown in Figure 1, which shows the absolute values of log likelihoods for each of the three categories. In this graph, a **lower** absolute log-likelihood indicates better model fit. For each category, the graph shows three bars, the first showing the log likelihood for the given category's sequences under the *Learner* model, the second bar showing the log likelihood for the given category's sequences under the *Prior Mastery* model, and the third bar showing the log likelihood for given category's sequences under the *Non-Learner* model. We conducted a series of paired *t*-tests to determine, for each group, whether there were significant differences between the log likelihoods for its own model and those for the other two models. For the *Learner* group model, its own log likelihoods were found to be significantly better than the log likelihoods of the other two models at the $p < .01$ level. For both of the other two models, *Prior Mastery* and *Non-Learner*, their own log likelihoods were found to be significantly different than the other respective models with even greater significance, at the $p < .001$ level.

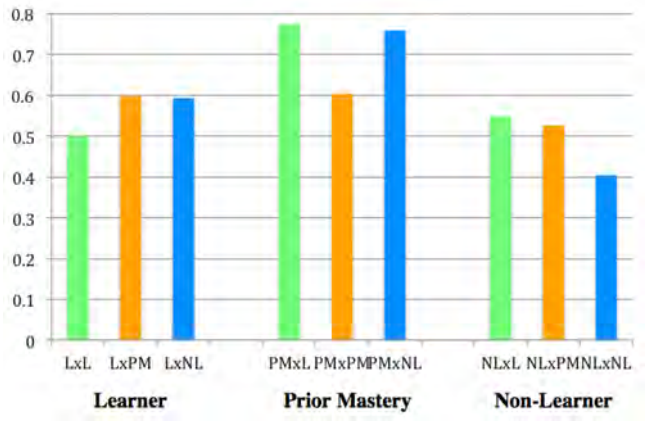


Figure 1. Absolute value of log likelihoods for each of the three categories. Lower values indicate better model fit.

Finally, we investigated the extent to which these models could classify *Learner, Prior Mastery*, and *Non-Learner* based only on the observed gameplay sequences in Room 2 and using leave-one-out cross-validation. A sequence was labeled with the group whose model produced the highest log-likelihood for that sequence (using only models that were trained with the sequence left out). Using this classifier, for the *Learner* category, 89.5% of pairs (17 out of 19) were correctly classified. For the *Prior Mastery* category, 100% of pairs (23 out of 23) were correctly classified. For the *Non-Learner* category, 95% (19 out of 20) were correctly classified. On the whole, this reflects a 95.2% accuracy in classifying whether a collaborative pair of students would be in the *Learner*, *Prior Mastery*, or *Non-Learner* group.

4. CONCLUSION

Modeling collaborative learning is an important direction for educational data mining research. We have demonstrated that sequence modeling relying on first-order Markov models can differentiate gameplay sequences of pairs where at least one partner learned from pairs who did not learn. Moreover, these models can classify those gameplay sequences with very high accuracy according to whether the dyad learned or not.

The opportunities are numerous for empirical studies into collaborative gameplay, problem solving, and dialogue. For example, the current analysis assumes that the maximal knowledge of the group is expressed through gameplay, an assumption that needs to be investigated. Additionally, a natural next step is to examine prediction power of individual learning along with the slightly more abstracted dyadic learning considered here. It is hoped that this line of investigation will move us toward highly effective support of dyadic learning.

5. REFERENCES

[1] Buffum, P.S. et al. 2016. Collaboration and Gender Equity in Game-Based Learning for Middle School Computer Science. *Computing in Science & Engineering*. 18, 2 (Mar. 2016), 18–28.

[2] Coleman, B. and Lang, M. 2012. Collaboration Across the Curriculum: A Disciplined Approach to Developing Team Skills. *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE '12)* (2012), 277–282.

[3] Falkner, K. et al. 2013. Collaborative Learning and Anxiety: A phenomenographic study of collaborative learning activities. *Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)* (2013), 227–232.

[4] Hickey, D.T. et al. 2009. Designing Assessments and Assessing Designs in Virtual Educational Environments. *Journal of Science Education and Technology*. 18, 2 (Feb. 2009), 187–208.

[5] Rafferty, A. et al. 2013. Estimating Student Knowledge from Paired Interaction Data. *Proceedings of the 6th International Conference on Educational Data Mining* (2013).

[6] Richard, G.T. and Hoadley, C. 2015. Learning Resilience in the Face of Bias: Online Gaming, Protective Communities and Interest-Driven Digital Learning. *Proceedings of the 11th International Conference on Computer Supported Collaborative Learning* (2015), 451–458.

[7] Stahl, G. et al. 2006. Computer-supported collaborative learning: An historical perspective. *Cambridge handbook of the learning sciences*. (2006), 409–426.

[8] Warren, S.J. et al. 2008. A MUVE Towards PBL Writing. *Journal of Research on Technology in Education*. 41, 1 (Sep. 2008), 113–140.

Can Word Probabilities from LDA be Simply Added up to Represent Documents?

Zhiqiang Cai
University of Memphis
Memphis, TN, USA
zcai@memphis.edu

Haiying Li
Rutgers University
New Brunswick, NJ, USA
haiying.li@gse.rugers.edu

Xiangen Hu
University of Memphis
Memphis, TN, USA
xhu@memphis.edu

Art Graesser
University of Memphis
Memphis, TN, USA
agraesser@memphis.edu

ABSTRACT
This paper provides an alternative way of document representation by treating topic probabilities as a vector representation for words and representing a document as a combination of the word vectors. A comparison on summary data shows that this representation is more effective in document classification.

Keywords
Topic modeling, LDA, document clustering, cluster similarity

1. INTRODUCTION
Topic modeling has been one of the most important methods in natural language analysis. It helps to discover underlying topics in a collection of documents. The found topics are used to form topic features for documents. The topic features are then used as input to perform task such as document clustering [11], automated summarization [1], automated essay grading [6], etc. LDA (Latent Dirichlet Allocation) [2, 3] is the most popular way for topic modeling. LDA topic model provides topic proportions as a vector representation of document. We investigated an alternative way of document representation by summing up word probabilities from LDA topic model. The new representation is compared with the topic proportion representation as input of a document clustering task on a summarization data set. The results showed that the simple "probability sum" document representation performs better.

2. LDA and Document Representations
Latent Dirichlet allocation (LDA), first introduced by Blei, Ng and Jordan in 2003 [3], is one of the most popular methods in topic modeling. LDA represents topics by word probabilities. Given a vocabulary with N words, $\{w_1, w_2, \cdots, w_N\}$, the LDA model probabilities $\boldsymbol{P}_k = (p_k(w_1), p_k(w_2), \cdots, p_k(w_N))$ form a representation of the k^{th} topic $(k = 1,2,\cdots,K)$. The words with highest probabilities in each topic usually give a good idea about what the topic is.

In LDA, a document d has an inferred topic proportion which is usually used as topic features to represent the document:

$$\boldsymbol{T}(d) \sim (t_1(d), t_2(d), \cdots, t_K(d)).$$

From the point of view of statistics, topic proportion is probably the only choice for LDA-based document representation. However, if we jump out of the box of statistics, we can simply view the word probabilities across the K topics as a K -dimensional vector representation for each word. Thus, a document can be represented by summing up the word probability vectors:

$$s_k(d) = \sum_{i=1}^{N} p_k(w_i) \log(1 + f(w_i, d)), (k = 1, 2, \cdots, K)$$

In the above formula, $s_k(d)$ is the "probability sum" of the document d on the k^{th} topic, $p_k(w_i)$ is the probability of the word w_i on the k^{th} topic, and $f(w_i, d)$ is the frequency of the word w_i in the document d. The logarithm of word frequency is known as Zipf scale [9].

3. Corpus for Document Clustering
201 participants wrote 1481 summaries for 8 passages, about 185 for each passage [10]. The lengths of the passages ranged from 195 to 399. The Flesch-Kincaid grade level was from 8.6 to 11.7. Some passages had similar topics: *Working and Running*, *Kobe and Jordan,* and *Effects of Exercising* on sports and exercising; and *Floods* and *Hurricane* on disasters.

The summaries were collected from an online experiment. The original goal was to evaluate the effect of an online AutoTutor [5, 9] lesson that teaches summarization. Each subject composed summaries for 2 texts before learning the lesson, 2 after learning, and 4 during learning with a counter-balanced design. The participant wrote each summary immediately after reading a passage. The system automatically controlled summary length (50-100 words) and *plagiarism*. The summary could not be submitted when it was out of range or when it had 10 consecutive words copied from the original passage.

Each summary was treated as a document for topic modeling. The vocabulary size was 4275 after removing stop words. 6 topic models were built for different numbers of topics (4, 8, 12, 16, 20 and 24), respectively. For each model, the topic proportions and the probability sums were computed for each summary. The LDA package used for topic modeling was infer.net from Microsoft [8].

Topic proportions and probability sums were then used as document features for clustering. We used K-Mean clustering method and fixed the number of clusters to 8 for all 6 topic models.

4. Results
We define the similarity of two clustering results by

$$Sim = \frac{\sum_{i=1}^{C} number\ of\ shared\ documents\ in\ cluster\ pair\ i}{total\ number\ of\ documents}.$$

The cluster pairs were best arranged using "Hungarian Algorithm" [7] so that the similarity is the highest under the paring. For each of the two document representations, we first compared the cluster similarity between models with the number of topics 4 and 8, 8 and 12, 12 and 16, 16 and 20, and 20 and 24. We aimed to check whether or not the clusters converge as the number of topics increases.

The results showed that when the number of topics increased, clustering based on probability sum quickly converged. The similarity between 12 topics and 16 topics was 0.96. For topic-proportion-based clustering, the similarity between 8 and 12 topics went close to probability sum. However, it dropped at 12 and 16, and then went up to 0.81 for 20 and 24.

While both representations converged to some clusters, the topic-proportion-based clustering converged to the unevenly distributed clusters. The largest two clusters contained 908 documents out of 1480. In contrast, probability-sum-based clustering converged to clusters of sizes almost the same as the original summary groups.

Table 1 shows the best matched clusters to the original passages for 24-topic model. Topic-proportion-based clusters matches the original passage groups with a similarity of 0.60, whereas probability-sum-based clustering did surprisingly better. The cluster similarity to the original summary grouping was 0.98.

Table 1 Best matched clusters to original passages

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
	Topic Proportion Based Clusters							
BM	160	0	0	0	0	20	1	2
Di	6	5	101	1	0	69	0	0
EE	0	1	186	0	1	1	0	0
Fl	11	7	21	1	1	139	5	1
Hu	1	0	1	1	173	3	5	0
JM	0	0	1	0	0	179	0	1
KJ	0	0	0	0	1	1	185	1
WR	1	0	164	0	1	20	0	1
	Probability Sum Based Clusters							
BM	180	0	0	1	0	1	1	0
Di	0	176	0	0	0	6	0	0
EE	0	1	182	0	0	5	1	0
Fl	0	0	0	179	1	6	0	0
Hu	0	0	0	0	180	4	0	0
JM	0	1	0	0	0	179	1	0
KJ	0	0	0	0	1	1	186	0
WR	0	0	2	0	0	4	0	181

Note: **BM**=Butterfly and Moth, **Di**=Diabetes, **EE**=Effects of Exercising, **Fl**=Floods, **Hu**=Hurricane, **JM**=Job Market, **KJ**=Kobe and Jordan and **WR**=Working and Running.

The cluster similarity changed when the number of topics increased in topic modeling. The topic-proportion-based clustering had its highest cluster similarity 0.77 to the original grouping when the number of topics is 12. It then dropped below 0.60. The probability-sum-based clustering had higher similarities for all models than topic proportion and consistently converged toward 1.

5. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation (DRK-12-0918409, 1108845), the Institute of Education Sciences (R305H050169, R305B070349, R305A080589, R305A080594, R305G020018, R305C120001), Army Research Lab (W911INF-12-2-0030), and the Office of Naval Research (N00014-00-1-0600, N00014-12-C-0643). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF, IES, or DoD.

6. REFERENCES

[1] Arora, R. and Ravindran, B. 2008. Latent Dirichelet allocation based multi-document summarization. In *Proceedings of the second workshop on Analytics for noisy unstructured text data* (Singapre, July 24 - 24, 2008). ACM, New York, NY, 91-97.

[2] Blei, D., Griffiths, T., Jordan, M., and Tenenbaum, J. 2004. Hierarchical topic models and the nested Chinese restaurant process. *Advances in Neural Information Processing Systems*, 16 (2004).

[3] Blei, D. M., Ng A. Y., and Jordan M. I. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3 (March, 2003), 993-1022.

[4] Chang, J., Gerrish, S., Wang, C., Boyd-Graber, J. L., & Blei, D. M. 2009. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*. (2009). 288-296.

[5] Graesser, A. C., D'Mello, S. K., Hu, X., Cai, Z., Olney, A., and Morgan, B. 2012. AutoTutor. In P. M. McCarthy, & C. Boonthum (Eds.), *Applied natural language processing and content analysis: Identification, investigation and resolution*. Hershey, PA: IGI Global. 169-187.

[6] Kakkonen, T., Myller, N., and Sutinen, E. 2006. Applying latent Dirichlet allocation to automatic essay grading. In *Advances in Natural Language Processing*. Springer Berlin Heidelberg, 110-120.

[7] Kuhn, H. W. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*， 2 (1955), 83–97.

[8] Lau, J. H., Grieser, K., Newman, D., & Baldwin, T. 2011. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* 1 (June, 2011). Association for Computational Linguistics. 1536-1545.

[9] Li, H. (2015). *The impact of pedagogical agents' conversational formality on learning and learner impressions* (Unpublished doctoral dissertation). University of Memphis, Memphis.

[10] van Heuven, W.J.B., Mandera, P., Keuleers, E., and Brysbaert, M. 2014. Subtlex-UK: A new and improved word frequency database for British English. *Quarterly Journal of Experimental Psychology*, 67 (2014), 1176-1190.

[11] Xie, P. and Xing, E. 2013. Integrating document clustering and topic modeling. In *Proceedings of the Twenty-Ninth Conference Annual Conference on Uncertainty in Artificial Intelligence* (Bellevue, Washington, USA, July 11 - 15, 2013). UAI 2013. AUAI, Corvallis, Oregon, 694-703.

Examining the necessity of problem diagrams using MOOC AB experiments

Zhongzhou Chen
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, MA, 02139
617-324-2731
zchen22@mit.edu

Neset Demirci
Balıkesir Üniversitesi
Bigadiç Cd., 10145 Paşaköy
Balıkesir, Turkey
0.266.2412762
ndemirci@gmail.com

David Pritchard
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, MA, 02139
617-253-6812
dpritch@mit.edu

ABSTRACT
Earlier research on problem solving suggested that including a diagram in a physics problem brings little, if any, benefit to students' problem solving success. In 6 AB experiments conducted in our MOOC, we tested the usefulness of problem diagram on 12 different physics problems, collecting over 8000 student responses in total. We found that including a problem diagram that contains no additional information very slightly improves the first attempt correct rate. On the other hand, in half of the cases, removing the diagram significantly increased the fraction of students who elected to draw their own diagram during problem solving. The results suggest that in contrast to conventional wisdom, the benefit of including a problem diagram rarely justifies the cost of creating one.

Keywords AB experiments, MOOC, problem diagrams.

1. INTRODUCTION
As instructors, we often feel obliged to accompany the problems we write with a figure or a diagram, even when all the necessary information is already included in the problem body. However in many cases, creating a "good looking" diagram or figure can be significantly time consuming and expensive. Therefore, it is a valuable question to ask whether a problem diagram does indeed help students solve problems more accurately or more quickly, and if so, does the benefit justify the cost of creating one?

Cognitive learning theories, such as dual coding hypothesis [7] and multimedia learning theories [6, 8] indirectly suggest, that diagrams can be potentially beneficial to problem solving. On the other hand, a series of recent experiments by Lin, Maris and Sigh[2–4]found that for the problems involved in their study, the accompanying diagrams have no detectable benefit for problem solving, and sometimes hurt performance by discouraging students to draw their own diagrams during problem solving.

Using the "split test" feature of the edX platform [1], this study addresses the following research questions in the context of a calculus based introductory mechanics course:

1. Do diagrams in general have an impact on students' problem solving performance (either percentage of correct answer or time spent on problem solving)? If so, to what extent?

2. Do diagrams change students' problem solving behavior, or more specifically, their decision to draw their own diagram?

2. MATERIALS AND METHODS

2.1 AB experiment on the edX platform
The edX platform allows the course creator to create controlled AB experiments by splitting the student population into two or more groups (called "partitions"), and presenting each group with a different version of content, such as a problem or a series of problems and html pages. Every student who tries to access the experimental course content for the first time is randomly assigned to one of the groups at the time of the access.

2.2 Experiment Design
A total of six experiments with identical design were implemented throughout the first eight units of the course. Each experiment involves two problems chosen from either the homework or the quiz section of a given unit, so the study involves twelve different problems in total. The problems were chosen from the first eight units of the course, covering kinematics, Newton's laws, circular motion, conservation of momentum, and conservation of energy.

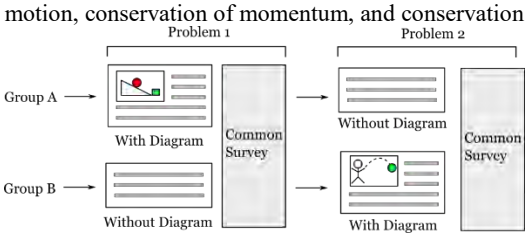


Figure 1: Experiment design. Each experiment consists of a pair of problems differing only in whether (DG) or not (NDG) they had a diagram. The same design is used for all 6 experiments conducted.

In each two-problem experiment, the student population was randomly partitioned into two groups: A and B (**Figure 1**). Group A saw the first problem in DG format and the second problem in NDG format. Group B saw the two problems in the same order, but the DG/NDG condition was reversed. The group assignment for each experiment is independent, reducing systematic bias.

Depending on when each experiment was released to students in the course, the number of students in each group ranged from ~480 (week 2) to ~180 (week 7).

The following survey question was asked after each problem:

When solving this problem, (check all that apply)

- o I drew one or more diagrams
- o I wrote down some equations
- o I did the problem entirely in my head
- o I used some other means to solve the problem

Only students who answered both the problem and the survey were included in the analysis.

3. RESULTS AND DISCUSSION

3.1 Results

We first look at the impact of including a diagram on the percentage of correct answer on students' first attempt. In most cases (see Fig 2 below) the presence or absence of a diagram has little impact on the difficulty of the problem itself. Only 3 out of 12 problems (P3, P4 and P8) showed a significant difference in difficulty between the two conditions ($p < 0.05, \chi^2 > 5$).

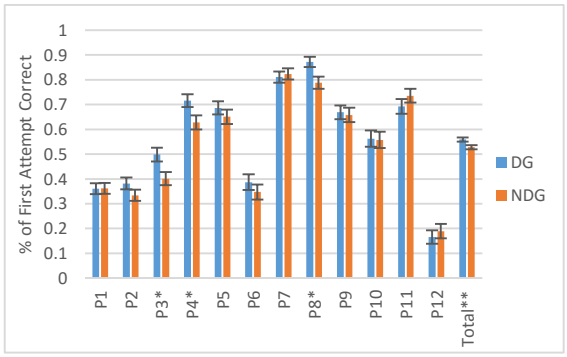


Figure 2: Percentage of first attempt correct for each problem. *Difference is significant at the 0.05 level. ** Difference is significant at the 0.01 level. (Chi-squared test)

Since we carefully balanced systematic bias in the population in our experiment design, it is meaningful to add up the data from all 12 problems and compare the overall success rate between the DG vs. NDG conditions (rightmost column in Fig 2). The overall correct rate under the DG condition is higher than that in the NDG condition by 3 ± 0.8 %. The difference, although small, is still statistically significant due to the large cumulative sample size (~ 3500 observations per condition, $p < 0.01, \chi^2 = 6.9$).

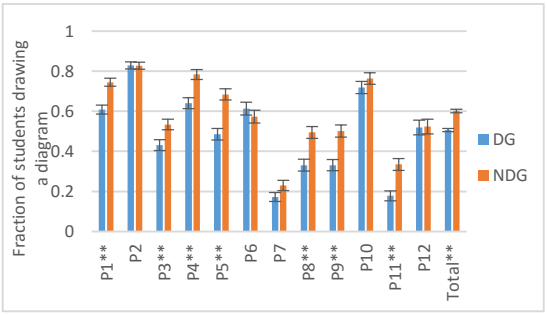


Figure 3: Percentage of students who drew a diagram solving each problem. *Difference is significant at the 0.05 level. ** Difference is significant at the 0.01 level. (Chi-squared test)

The presence/absence of a problem diagram impacts students' tendency to draw their own diagram as measured by the survey question. As shown in Figure 3, on 7 out of 12 problems, a significantly lower fraction of students ($p < 0.01, \chi^2 > 7$, Chi-square test) in the DG condition reported drawing their own diagram during problem solving than in the NDG condition. A noteworthy observation (Fig. 3) is the high variation in sensitivity of different problems to the DG/NDG condition. Combining the data across all 12 problems, students in the DG condition are 10% less likely to draw their own diagram than in the NDG condition ($p < 0.001, \chi^2 = 65$).

3.2 Discussion

Perhaps the most surprising observation of this study is how little students benefit from a problem diagram. Even with the large sample size provided by MOOC, significant difference between the two conditions are only observed for 3 out of 12 problems, with the largest difference at 10% and the overall difference at merely 3%.

Those results suggest that even though the benefits predicted by conventional wisdom and dual-coding hypothesis may still exist, the effect size might be small in an *in vivo* situation and only significant in the more extreme cases. For the majority of "normal" physics problems, our findings are consistent with previous studies [2–5] indicating that the benefit of a diagram is small.

In stark contrast to the correct rate, the decision to draw is very sensitive to the DG/NDG condition on 7 out of 12 problems: when the problem diagram is removed, students are 10% more likely to draw their own.

For instructors, the study suggests that for common physics problems of average difficulty, the benefit of adding a diagram may be too small to justify the resource and effort required to create it.

4. ACKNOWLEDGMENTS

Our thanks to Dr. Qian Zhou for helping on data analysis.

5. REFERENCES

[1] edX Documentation: Creating Content Experiments: *http://edx.readthedocs.org/projects/edx-partner-course-staff/en/latest/content_experiments/index.html*.

[2] Lin, S.-Y. et al. 2013. Student difficulties in translating between mathematical and graphical representations in introductory physics. 250, (2013), 250–253.

[3] Maries, A. and Singh, C. 2014. A good diagram is valuable despite the choice of a mathematical approach to problem solving. *2013 Physics Education Research Conference Proceedings*. (Feb. 2014), 31–34.

[4] Maries, A. and Singh, C. 2012. Should students be provided diagrams or asked to draw them while solving introductory physics problems? *AIP Conference Proceedings*. 1413, (2012), 263–266.

[5] Maries, A. and Singh, C. 2013. To use or not to use diagrams: The effect of drawing a diagram in solving introductory physics problems. *AIP Conference Proceedings*. 1513, 1 (2013), 282–285.

[6] Mayer, R.E. 2001. *Multimedia Learning*. Cambridge University Press.

[7] Paivio, A. 1986. *Mental representations: a dual coding approach*. Oxford University Press.

[8] Schnotz, W. 2002. Towards an Integrated View of Learning From Text and Visual Displays. *Educational Psychology*. 14, 1 (2002), 101–120.

Identifying relevant user behavior, predicting learning, and persistence in an ITS-based afterschool program

Scotty D. Craig
Arizona State University
scotty.craig@asu.edu

Xudong Huang
The University of Memphis
xhuang3@memphis.edu

Jun Xie
The University of Memphis
jxie2@memphis.edu

Ying Fang
The University of Memphis
yfang2@memphis.edu

Xiangen Hu
The University of Memphis
xhu@memphis.edu

ABSTRACT

ALEKS (Assessment and Learning in Knowledge Spaces) has recently shown promise for effectively training mathematics at equivalent levels to human teachers. However, not much is known about how the system accomplished this. In this paper, we describe the use of three data mining techniques used to analyze student data from an afterschool program with ALEKS. Our first analysis used DMM modeling and k-clustering to identify important groups of behaviors within ALEKS users and to show the importance of context for elements. Our second analysis focused on identifying learner behaviors that predict student learning during the program. The final analysis presents a method for determine learner persistence within the afterschool program.

Keywords

ALEKS, Afterschool programs, learning strategies, help seeking, persistence

1. INTRODUCTION

ALEKS is a web-based learning system with artificial intelligence components that are based in Knowledge Space Theory [1]. Instead of giving scores to measure a student's overall mastery of the subject, the theory allows for a precise assessment of what the student knows, does not know, and is ready to learn next. The probability of mastery for a knowledge state increases as students correctly answer questions containing that problem type.

ALEKS is a highly effective educational technology program shown to perform at the same level as other major ITS systems in mathematics [2]. In a four year evaluation of ALEKS in an afterschool setting, the students tutored by ALEKS or taught by expert teachers in one after-school program showed the same level of performance in a mathematics state test [3,4], and outperformed controls not participating in the program[5].

1.1 Current investigation

1.1.1 ALEKS afterschool program

The afterschool program was implemented for 25-week after school. It was held twice a week for 2 hours each day. Students received three 20-minute learning segments with a 20-minute break between each. Student logs were recorded by ALEKS. The students were from five middle schools in west Tennessee. The schools were located in a mid-sized city and the surrounding rural area, having a largely economically disadvantaged population (68.2%) and large minority student enrollment (56.3% African American, 39.3% White, and 4.4% others). None of the five schools reach an average SES level of Tennessee (i.e., 54.4% of the students eligible for free or reduced-price lunch).

1.1.2 Research question

While the afterschool program demonstrated that students using ALEKS could perform at the same levels as student in teacher-led classrooms [3.5], the student's learning process that led to this result is still unclear. Summaries of three methods are presented to show how popular data mining techniques can be applied to ALEKS log files to better understand student's behavior in the ALEKS afterschool program.

2. Learning strategies with DMM

There are distinct advantages for analyzing sequences over raw frequencies. The frequency counts could indicate that the two students used the same strategy. However in context, the two students act differently because the patters have different sequences. Modeling learning sequences is not as direct as frequency counting. One way to measure sequence is to calculate similarities in sequences, and then cluster the sequences using the similarities. A method, modeling learning sequences with Discrete Markov Models (DMM) and clustering with a k-means algorithm, has successfully discovered help-seeking strategies in ITS [6].

The analysis used 55,281 learning sequences of 372 students on ALEKS system. Typical activities students made include: correct, wrong, explain, mastery (added to pie), failed, and left the attempt. We recoded the same actions in a row as action - action2 - action3 – action3 for easy interpretation.

With DMM modeling and k-means clustering for all transitions, ten learning strategies emerged. These strategies were Cluster 1 – three correct practices in a row and reach mastery (9%), Cluster 2 – Quick mastery (11%), Cluster 3 – keep practice after mastery (6%), Cluster 4 – Frequently request worked examples and only try when confident (7%), Cluster 5 – Request worked examples after wrong and get correct and mastery finally (12%), Cluster 6 – Request worked examples then quit without practice (13%), Cluster 7 – Request worked examples after wrong but still get wrong then quit (17%), Cluster 8 – Correct at 1st practice but wrong at 2nd & 3rd, then request worked examples but only get half practices correct then. (6%), Cluster 9 – All practice are wrong, request worked example after 2 wrongs, still get wrong, quit or reach failure. (9%), and Cluster 10 – All practice are wrong, reach failure and then 2nd failure (9%).

3. Learning behaviors and learning outcome

A sample from 204 students was used to predict students learning using behaviors within ALEKS. The learning behaviors recorded in ALEKS log files were categorized into help-seeking and practice. We utilized logistic mixed effects models to investigate the relationship of help-seeking and practice with learning outcome. Topics and students were random variables. The model also included student's pretest which was measured by 5th grade TCAP score. The learning outcome was topic mastery (1 or 0).

3.1 Help-seeking and learning outcome

The results of logistic mixed effects model indicated four significant help-seeking behaviors were predictive of learning (R^2 = .81, For full results See Table 1). We used 10-fold cross validation to validate the mixed effects model of help-seeking.

Table 1.
Student help-seeking behaviors that predict learning outcomes

Learning behaviors	Coefficient	Std. Err	z	p
Pretest	.35	.08	4.32	.000
Reading Explain first	.42	.14	3.12	.00
Proportion explain	-46.86	1.51	-31.13	.000
Explain after mistake	-.36	.35	-1.05	.29
Explain request latency	-.01	1.29	27.79	.000
Explain avoid mistake	35.99	.01	-2.40	.02

3.2 Practice and learning outcome

The results of logistic mixed effect model indicated five significant patters of making mistakes were related to learning (R^2 = .75, See Table 2 for results). A 10-fold cross validation was adopted to validate the mixed effects model of practice.

Table 2
Student practice behaviors that predict learning outcomes

Learning behaviors	Coefficient	Std. Err	z	p
Pretest	.17	.10	1.64	.10
Initial Mistake	.64	.09	7.23	.000
Mistake (%)	-5.35	.32	-16.85	.000
Success (%)	12.65	.49	26.04	.000
Self-correction	-1.3	.24	-5.52	.000
Self-correction time	.01	.003	2.23	.03

4. Prior knowledge, difficulty on persistence

A sample from 114 student log files utilizing 92,235 lines of log files data from years two and three of the program that included date, time, topics attempted and the result of each trial were used to predict student's persistence using prior knowledge topic difficulty and time period. The number of trials (T) was chosen as the measure of persistence. Then, three levels of persistence were defined: high persistence (T>15), medium persistence (10<=T<15), and non-persistence (T<5 and not reach mastery).

4.1 Results

Logistic regressions were performed to explore the effects of prior knowledge, topic difficulty and time period the learning took place on the likelihood of participant's persistence related behavior. For high persistence, the model was significant, $\chi2(3) = 124.14$, $p < .001$, explaining 2.8% (Nagelkerke R^2) of the variance of highly persistence students and correctly classified 96.2% of cases. For medium persistence, the model was significant, $\chi2(3) = 118.68$, $p < .001$, explaining 1.8% (Nagelkerke R^2) of the variance in medium persistence and correctly classified 93.3% of cases. Increasing topic difficulty was associated with increased persistence, but increasing prior knowledge and days learning in the system was associated with a reduction in persistence. For non-persistence, the model was statistically significant, $\chi^2(3) = 864.88$, $p < .001$, explaining 6.8% (Nagelkerke R^2) of the variance in non-persistence and correctly classified 62.5% of cases. Increasing topic difficulty was associated with an increased non-persistence. Increasing prior knowledge was associated with a reduction in non-persistence.

5. Discussion/conclusion

The current paper present three methods to analyze learner performance which identify important clusters of learner strategies during learning with ALEKS, help seeking behaviors that predict learning, and persistence. The first analysis clustered learner strategies and demonstrated that context is important when looking at clusters. Thus identical elements or techniques can serve different functions when the sequence occurs at a different point in the learning process. The second two analyses use features from the ALEKS data logs to predict learning and persistence. The second analysis found that latency to seek help was negatively related to mastering a topic. This is a validation that ALEKS is working in that increase practice with the system was predictive of mastery of topics. For student persistence, while predicted variability was small, the models were very reliable and able to classify a large proportion of the data. The pattern of data for non-persistent behavior was interesting finding that lower prior knowledge students work on problems projected to be of greater individual difficulty which is predictive of lower persistence. Taken together these techniques indicate patterns that are easily detected and corrected within systems like ALEKS.

6. ACKNOWLEDGMENTS

This research was supported by the Institute for Education Sciences (IES) Grant R305A090528.

7. REFERENCES

[1] Falmagne, J.C., Koppen, M., Villano, M., Doignon, J.P. and Johannesen, L., 1990. Introduction to knowledge spaces: How to build, test, and search them. *Psychological Review, 97*(2), 201.-224.

[2] Sabo, K.E., Atkinson, R.K., Barrus, A.L., Joseph, S.S. and Perez, R.S., 2013. Searching for the two sigma advantage: Evaluating algebra intelligent tutors. *Computers in Human Behavior*, 29(4), 1833-1840.

[3] Craig, S. D., Hu, X., Graesser, A. C., Bargagliotti A. E., Sterbinsky, A., Cheney, K. R., & Okwumabua, T. 2013. The impact of a technology-based mathematics after-school program using ALEKS on student's knowledge and behaviors. *Computers & Education*, 68, 495-504.

[4] Huang, X., Craig, S.D., Xie, J., Graesser, A. and Hu, X., 2016. Intelligent tutoring systems work as a math gap reducer in 6th grade after-school program. *Learning and Individual Differences*, 47, 258-265.

[5] Hu, X., Craig, S.D., Bargagliotti, A.E., Graesser, A.C., Okwumabua, T., Anderson, C., Cheney, K.R. and Sterbinsky, A., 2012. The Effects of a Traditional and Technology-based After-school Setting on 6th Grade Student's Mathematics Skills. *Journal of Computers in Mathematics and Science Teaching*, 31(1), 17-38.

[6] Vaessen, B.E., Prins, F.J. and Jeuring, J., 2014. University students achievement goals and help-seeking strategies in an ITS. *Computers & Education*, 72, 196-208.

Extracting Measures of Active Learning and Student Self-Regulated Learning Strategies from MOOC Data

Nicholas Diana
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
ndiana@cmu.edu

Michael Eagle
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
meagle@cs.cmu.edu

John Stamper
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
john@stamper.org

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
koedinger@cmu.edu

ABSTRACT
Previous work has demonstrated that in the context of Massively Open Online Courses (MOOCs), doing activities is more predictive of learning than reading text or watching videos (Koedinger et al., 2015). This paper breaks down the general behaviors of reading and watching into finer behaviors, and considers how these finer behaviors may provide evidence for active learning as well. By characterizing learner strategies through patterns in their data, we can evaluate which strategies (or measures of them) are predictive of learning outcomes. We investigated strategies such as page re-reading (active reading) and video watching in response to an incorrect attempt (active watching) and found that they add predictive power beyond mere counts of the amount of doing, reading, and watching.

Keywords
MOOCs, active learning, self-regulated learning

1. INTRODUCTION
The growing popularity of MOOCs has prompted an examination of the effectiveness of prototypical MOOC activities such as watching video lectures. Most recently, Koedinger et al. (2015) explored the impact of watching video lectures, reading course content, and doing interactive activities. They found that doing activities had a larger impact than reading course content or watching videos. The authors attribute this effect, at least in part, to the fact that doing activities is necessarily an active form of learning, whereas reading content and watching videos is generally passive.

However, not all *reading* and *watching* is done passively. This study returns to the dataset used in Koedinger et al. (2015) and attempts to extract new features that are representative of different types of active learning behaviors and student strategies. By exploring these finer-grained measures of student behavior, we are able to: 1) support the results of Koedinger et al. (2015) by providing more evidence that active learning behaviors are associated with better learning outcomes, and 2) demonstrate that evidence of active learning can not only be mined from *doing* data, but from *reading* and *watching* data as well.

2. BACKGROUND
2.1 Previously Explored Features
Koedinger et al. (2015) designed three features to capture *doing, watching*, and *reading* behavior within a MOOC. *Doing* behavior was characterized by the total number of activities started throughout the course. *Watching* behavior was characterized by the number of times the user clicked play while viewing a video in the MOOC (referred to by the feature name "video"). In this count, consecutive plays of the same video were not counted.

The course content and interactive activities often appeared on the same page, so estimating a measure of *reading* behavior was slightly more complex. *Reading* was estimated using a ratio of about 3.4 activities per page, and then subtracting pages viewed for activity access from total pages viewed. While not as precise as some other measures, the goal of this measure is to capture variation in student reading.

Left unexplored are more complex features dependent on patterns of actions. We build off of the features previously explored in Koedinger et al. (2015) to generate features representative of student strategies embedded in watching and reading data.

2.2 Finer-grained Features
With respect to *watching* behavior, we extended beyond raw counts and instead looked at possible interactions between *watching* and *doing*. We hypothesized that students who complete problems while watching videos, and students who reference videos after incorrect attempts do better on the final exam. For *reading* behavior, we examined the impact of the common, albeit surface-level strategy of reviewing a page to re-read content [1,2], hypothesizing that students who review content do better on the final exam.

3. DATA AND METHOD
3.1 Data
The data used are from a 12-week survey course titled "Introduction to Psychology as a Science." The lectures, along with slides, a discussion form, quizzes, and exams, were provided via Coursera. The Open Learning Initiative (OLI) Learning Environment was embedded into Coursera to provide readings and interactive activities.

The current study used a subset of this dataset, which contains only students who registered for the OLI portion of the course and took the final exam (N=939). On average each student generated 2757 transactions, though the actual number varied greatly among students (SD=1909). This dataset is freely available (with administrator permission) via the online learning data repository and analysis service, DataShop [3] at:

https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=863.

3.2 Model Building
To understand the impact of the new features on learning outcomes relative to the previously explored features, a linear regression model was generated that included the three original *watching, reading,* and *doing* features. This model serves as a baseline. A new linear model was generated for each new feature. The new feature was added alongside the previously explored features to predict final exam score, unless it was redundant with another feature.

4. RESULTS AND DISCUSSION

4.1 Baseline Model

As expected, the baseline model showed that the *doing* measure had a high impact on final exam performance ($p<.001$). Neither the *reading* nor the *watching* measures were significant predictors. The results of this model can be seen in Table 1 in the row labeled "Baseline."

4.2 Watching

4.2.1 Attempting Activities During Video Playback

We hypothesized that some students may be watching videos and doing activities simultaneously, potentially answering questions as the relevant material is covered in the video lecture. To test this hypothesis, we extracted a new feature that represents the proportion of all activity attempts that occurred during video playback. When added to the baseline model, the proportion of attempts that occurred during video playback was predictive of final exam performance, though marginally significant ($p<.1$). This may indicate that some students are answering problems while watching relevant videos, and that this is a successful strategy. The results of this model can be seen in Table 1 in the row labeled "% attempts during playback."

4.2.2 Referencing Videos After Incorrect Attempts

We similarly hypothesized that some students may reference the video lectures after an incorrect attempt on an activity. To test this, we extracted a new feature representing the proportion of all video play actions that occurred after an incorrect attempt, but before the next attempt on the same problem. When added to the baseline model, the proportion of video play actions that occurred between attempts on the same problem was predictive of final exam performance, though again, marginally significant ($p<.1$). This may indicate that some students are referring back to videos to find correct answers. The results of this model can be seen in Table 1 in the row labeled "% plays after incorrect attempts."

4.3 Reading

4.3.1 Only-Reading Page Views

In the current version of OLI course content and activities appear on the same page. To compensate for this, we counted the number of pages viewed without any activity attempts. To mitigate pages viewed quickly on the way to another page, we eliminated any page viewed less than 10 seconds from this count. When added to the baseline model (with "non-activity page views" removed for redundancy), the number of only-reading page views is predictive of final exam performance ($p<.05$). The results of this model can be seen in Table 1 in the row labeled "Only-reading page views." Note that this is by no means a complete measure of all *reading* behavior because it misses any reading done on pages where the student also attempted activities.

4.3.2 Re-reading Page Views

We also found that, when added to the baseline model (again with "non-activity page views" removed for redundancy), the number of second page views that are reading only page views (i.e., pages revisited with 0 activity attempts) is predictive ($p<.001$). This suggests at least some students review material by re-reading course content, and that this strategic reading is predictive of final exam performance. The results of this model can be seen in Table 1 in the row labeled "pages re-read."

5. CONCLUSION

Our work examines how evidence of active learning can be extracted from *reading* and *watching* data as well as *doing* data, and demonstrates that these measures can be predictive of learning outcomes. Re-reading pages (a measure of active reading) and attempting activities while watching videos (active watching) improved prediction of learning outcomes beyond the simple measure of active doing. While more research is needed to test their generality, these features may help establish a more nuanced characterization of learner strategies.

6. REFERENCES

[1] Alexander, P. (1997). Mapping the multidimensional nature of domain learning: The interplay of cognitive, motivational, and strategic forces. In M. L. Maehr & P. R. Pintrich (Eds.), *Advances in motivation and achievement*, 213–250.

[2] Azevedo, R., Johnson, A., Chauncey, A., Graesser, A., Zimmerman, B., & Schunk, D. (2011). Use of hypermedia to assess and convey self-regulated learning. *Handbook of self-regulation of learning and performance*, 102-121.

[3] Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., & Stamper, J. (2010). A Data Repository for the EDM commuity: The PSLC DataShop. In Romero, Ventura, Pechenizkiy, Baker, (Eds.) *Handbook of Educational Data Mining*. CRC Press.

[4] Koedinger, K. R., Mclaughlin, E. a, Kim, J., Zhuxin Jia, J., & Bier, N. L. (2015). Learning is Not a Spectator Sport: Doing is Better than Watching for Learning from a MOOC, L@S (Mar. 2015), 111–120. DOI=http://doi.org/10.1145/2724660.2724681

Table 1. Linear regression models that include new features.

Added Feature	Activities Started	Non-Activity Page Views	Video	Added Feature(s)	RMSE	Adj. r^2	AIC
N/A (baseline)	1.8206***	0.3632	0.1509	-	6.768	0.0785	6261.855
% attempts during playback	1.8990***	0.2776	0.2241	0.3753.	6.472	0.0781	5541.207
% plays after incorrect attempts	1.9263***	0.2653	0.1361	0.3845.	6.66	0.0811	5986.356
Only-reading page views	1.7775***	-	0.1458	0.5129*	6.759	0.0808	6259.458
Pages re-read	1.5436***	-	0.1437	0.8468***	6.736	0.0871	6253.016

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 .' 0.1 ' ' 1

Exploring Social Influence on the Usage of Resources in an Online Learning Community

Ogheneovo Dibie
Institute of Cognitive Science
Dept. of Computer Science
University of Colorado
Boulder, Colorado USA
ogdi0204@colorado.edu

Tamara Sumner
Institute of Cognitive Science
Dept. of Computer Science
University of Colorado
Boulder, Colorado USA
sumner@colorado.edu

Keith Maull
University Corporation for
Atmospheric Research
Boulder, Colorado USA
kmaull@ucar.edu

David Quigley
Institute of Cognitive Science
Dept. of Computer Science
University of Colorado
Boulder, Colorado USA
david.quigley@colorado.edu

ABSTRACT

This research investigates the usage distribution of instructional resources shared among educators in an online learning community. The usage of a resource is defined by the number of unique educators who use (click on) it. We explored what the usage distribution of these resources looks like and we investigated what underlying mechanisms may have generated the observed distribution. Our results indicate that the usage distribution of resources follows a power law. Furthermore, our results also suggest that an educator's decision to use a resource may be influenced by the prior decisions of others. 82.6% of 2500 simulations of an information cascade model developed to model the resource selection process of educators resulted in a power law distribution as observed in our data. Information cascades provide a natural way of understanding how individuals may imitate the decisions of others even when such decisions do not align with their perssonal preferences.

1. INTRODUCTION

Research consistently indicates that online learning communities can improve the instructional practices of educators and produce increases in student learning outcomes by providing educators with access to learning resources and best practices shared by their peers [5]. Given the importance of these community-contributed resources to educator instruction, understanding the factors that encourage their usage is an intriguing question with important implications for educator instruction, student learning and agencies that support these communities.

We explored this question in the context of a community of Earth Science educators that used an online curriculum planning tool called the Curriculum Customization Service (CCS). The CCS provides educators with access to digital versions of their class textbook, digital library resources and community-contributed resources. This study is based on 6th-9th grade Earth Science educators that shared and used community-contributed resources in the CCS over a period of four academic years.

We began by exploring the observed usage distribution of community-contributed resources in the CCS, and then turned our attention to the influence of three mechanisms on the observed usage distribution. First, we investigated how resource visibility influences resource selection—postulating that the position or rank of a resource in the list it is displayed in *may* impact selection behavior. Second, we investigated how the quality (or perceived quality) of a resource might have influence on selection. Finally, we examine how social factors, specifically how the decisions of others in the community, provide insights into the observed resource usage distribution.

2. METHODS AND RESULTS

We discovered that the usage distribution of community-contributed resources follows a power law. Also known as Zipf, Pareto-Levy or scale-free distributions [4], a quantity x obeys a power law if it is drawn from a probability distribution $p(x) \propto x^{-\alpha}$ where α is known as the exponent or scaling parameter. Power laws appear in a wide array of man made and natural phenomena [3] such as the distribution of calls to telephone numbers, scientific paper citations and the frequency of use of English words [4]. We determined that the usage distribution of resources followed a power law using software implementations[1] [2] of the rigorous statistical approach of Clauset et al. [3] for detecting power laws in empirical data. [3]. Our empirical data was found to follow a power law with an α of 4.44 and an x_{min} value of 15. Figure 1 illustrates that a power law provides a closer fit to the complimentary cumulative distribution function (CCDF) [3]

[1] plfit: https://pypi.python.org/pypi/plfit
[2] powerlaw: https://pypi.python.org/pypi/powerlaw
[3] The CCDF is defined by $\Pr(X \geq x)$

of the empirical data in comparison to the lognormal and exponential distribution.

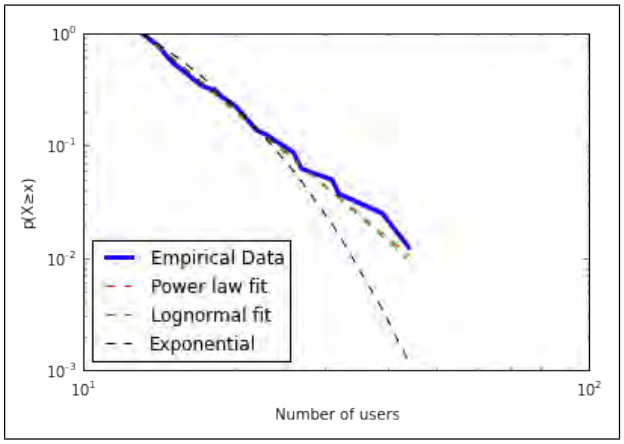


Figure 1: Comparisons of the complimentary cummulative distribution function (CCDF) of the empirical data, the power law, lognormal and exponential distribution fits to the data.

2.1 Mechanisms behind the power law distribution of resources

Resource position: Correlation tests between the mode, median and last click position of resources and their usage show only a very weak correlation between the usage of a resource and its position during the 2012-2013 school year. This suggests that a resource position had little to no influence on usage.

Resource quality: We then investigated the relationship between resource quality (inferred before a user clicks on it) and its usage in two steps. First, we used the presence of a description in the listing of a resource as a marker of its quality. Thus, resources with a description were deemed as having high quality and those without a description were regarded as low quality. We then investigated if there was a statistically significant difference in usage between resources of high quality and those of low quality, and consequently discovered no statistically significant difference in usage between resources of both groups. Our next investigation into the impact of a resource's quality on usage investigated whether there was any correlation between the number of quality signals of a resource and its usage. To do this, we developed a composite resource quality score that incorporated all signals of a resource's quality that can be inferred by a user before clicking. These signals were mapped to the resource quality indicators developed by Bethard et al. [1]. Our results show only a weak correlation of 0.124 between resource quality and usage ($t = 2.8343, df = 516, p = 0.002387$)

Social influence: Finally, we looked at the impact of aggregate social influence on the usage of community-contributed resources. We found a statistically significant positive correlation of 0.634 at a p-value of $2.2e^{-16}$ between saves and usage. Unlike our earlier tests on position and quality, this indicates that the social influence conveyed through the saving of resources may be in part responsible for driving usage.

We then explored if an information cascade model simulating the decision making processes of educators can generate a power law usage distribution as observed in our data. Our model extends the informational cascade model of Bikchandani, Hirshleifer, and Welch (BHW) [2] in three ways. First, instead of the binary decision model of BHW, a decision will be made between $1..r$ resources at any time. Second, in contrast to the BHW model, the decision of an individual is not always visible to others as a public signal. In our context, the only public signal available is whether or not a user saves a resource. After clicking on a resource, users will leave public signals with a uniform random probably p. This probability is exogenously fixed at 0.41—determined from computing the ratio of saves to unique clicks on all resources across all school years. Finally, a user's private signal p_s for a resource r is drawn from a discrete uniform probability distribution such that $p_s \in [0, 1]$. 2500 simulations of the information cascade model described above were processed with each simulation evaluated to see if they follow a power law using the procedure of Clauset et al. [3]. Consequently, 82.6% of these simulations were determined to follow a power law distribution. The outcomes of this experiment strongly suggests that an information cascade model simulating the decision making process of educators can lead to a power law usage distribution as observed in our data. This provides strong support for the social influence hypothesis as a generative mechanism for the observed usage distribution.

3. DISCUSSION & CONCLUSION

For agencies that support online learning communities, this research has important implications for resource presentation and recommendation. In presenting resources, social influence signals can be de-emphasized to limit the chances that they will detract users from evaluating a resource's inherent quality. For example, in the CCS, the number of educators that have saved a resource can be hidden and require active effort from users to be revealed. In recommending resources, high quality but barely used resources can be recommended to educators in ways that give them high precedence. This could include personalized recommendations while active on the platform or email recommendations. This paper is based upon research supported by National Science Foundation awards #1043638 and #1147590.

4. REFERENCES

[1] Bethard, S., Wetzer, P., Butcher, K., Martin, J. H., and Sumner, T. Automatically characterizing resource quality for educational digital libraries. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries* (2009), ACM, pp. 221–230.

[2] Bikhchandani, S., Hirshleifer, D., and Welch, I. A theory of fads, fashion, custom, and cultural change as informational cascades. *Journal of political Economy* (1992), 992–1026.

[3] Clauset, A., Shalizi, C. R., and Newman, M. E. Power-law distributions in empirical data. *SIAM review 51*, 4 (2009), 661–703.

[4] Easley, D., and Kleinberg, J. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

[5] Louis, K. S., and Marks, H. M. Does professional community affect the classroom? teachers' work and student experiences in restructuring schools. *American journal of education* (1998), 532–575.

Time Series Cross Section method for monitoring students' page views of course materials and improving classroom teaching

Konomu DOBASHI
Faculty of Modern Chinese Studies
Aichi University
4-60-6 Hiraike-cho Nakamura-ku
Nagoya-shi Aichi-ken
453-8777 Japan
dobashi@vega.aichi-u.ac.jp

ABSTRACT
To enable teachers to monitor student engagement and improve classroom instruction, a data mining method and an Excel macro are developed in this work. The data mining method is based on a Time Series Cross Section (TSCS) framework and designed for application to students' page views of course materials that are created over Moodle. The Excel macro generates TSCS tables of students' page views and reflect the viewing behaviors of students over time as transitioning numerical values.

Keywords
Time Series, Cross Section, page views student engagement, educational data mining

1. INTRODUCTION
A teacher is responsible for ensuring proper delivery of lessons in the classroom while simultaneously understanding the individual reactions and progress of students. Effectively satisfying these roles are essential to improving the quality of education. The problem is that in a class comprising dozens of students, accurately measuring individual reactions and progress is difficult even for experienced teachers. Another challenge is how such data can be provided to both educators and learners. A favorable strategy is to supply teachers with the results of appropriately conducted analyses in a timely manner so that analytical insights can be used to advance teaching enhancement. A tool that can be employed frequently in class for such purpose is equally desirable. We propose an Excel macro that semi-automatically generates TCSC tables from Moodle logs. The system monitors and records the time that students spend on browsing and their page views in class. It also provides data and suggestions that can be used as reference for reinforcing classroom instruction and keeping track of student engagement.

2. RELATED RESEARCH
Currently, analyzing Moodle logs [6] is primarily based on Excel or CSV data. Because the macro developed in this study is grounded in Excel and pivot table functions, teachers can easily obtain the summaries of the frequency at which students view course materials [1, 2]. Moodog [7] that Zhang and Almeroth has developed that incorporated an analysis function of log in Moodle. This system is able to analyze the course materials browsing rate, page views and viewing time of students. The analytical results are displayed on the Moodle screens, it represents interaction of the students and Moodle using graphs and the tables.

Mazza and Dimitrova has been developed a system called CourseVis [3] that to track the student's behavior in an online class, it can be visualized by the graph along the access status to the content page to the course schedule. Also Gismo [4] also take advantage of the access history of Moodle and visualized using the access graph to the students of the courses and teaching materials, it is to understand the behavior of the students. In the current it has been provided so that it can be installed as part of the Moodle.

Google Analytics [2] provides a website analysis service that enables data analyses grounded in different perspectives. Such service also helps educators improve course materials and lessons. Whereas Google Analytics can be used only by a Moodle administrator, the method proposed in this paper can be employed by any Moodle user. The developed macro is equally accessible to any Moodle course administrator.

3. METHOD AND EXPERIMENTS
Excel has several features designed to process qualitative data. Among these, the pivot table feature enables users to count qualitative data, such as strings; create a cross section table; and quantify input data. These functions were applied in this work. The tabulation generated in this study is referred to as a "Time Series Cross Section table" because an aggregated pivot table was created to incorporate time series data into the analysis.

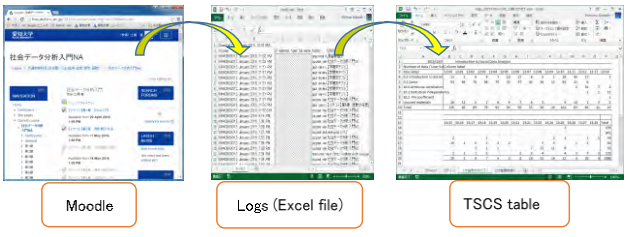


Figure 1. Overview of processing

This study primarily used PDF files that are viewable through a PDF viewer by clicking on a link in the table of contents created in the Moodle topics format, which is commonly used in Moodle based courses. The TSCS tables generated in this research features columns on time, user full name, action, and information. These

data are aggregated by using Excel's statistical functions and pivot table features to semi-automatically generate the TSCS tables.

While delivering a lesson, the teacher can assess student status and if necessary, download Moodle logs to a specified folder and run the macro. Downloading of logs and macro processing take only tens of seconds. These features guarantee that sufficient time and focus is devoted to a lesson. After a lesson is completed, the teacher can run the macro (if necessary) without having to worry about processing time during a class.

The developed macro was applied in the Introduction to "Social Data Analysis" class offered at the case university to demonstrate how a TSCS table is generated. Table 1 is the TSCS table of page views for course items (1-minute intervals). On December 9, 2015, the teacher discussed the lesson on attribute correlation for 90 minutes. The lesson was initiated at 13:00 and ended at 14:30. Table 1 shows the TSCS table generated at 1-minute intervals, downloaded at 13:28 from Moodle logs, and aggregated. Page views of the course items were counted from the beginning of the lesson up to 13:28 (Table 1).

The TSCS table for students (generated 2-minutes intervals) shows that viewing was concentrated from 13:12 to 13:16 and at 13:24 (Table 2). Some students exhibited a delay in accessing the materials at 13:18, 13:20, 13:26, and 13:28. With a TSCS table for each student, the teacher can determine which students are viewing materials and which have recently browsed the materials (Table 2).

Table 1. Example of TSCS table for course items generated 2-minutes intervals

Number of data / U	Column label																
Row label	13:02	13:04	13:06	13:08	13:10	13:12	13:14	13:16	13:18	13:20	13:22	13:24	13:26	13:28	13:30	Total
0.1 Introduction to Social Data	18	17	17	18	6	64	17	1					1			159
0.2 Quize	162	144	114	80	41	95	6									642
10.0 Attribute correlation						2	42	4		1			3	1	1	54
10.1 Statistical Independence							2	29	6	5	2	5	3	3		55
10.2 Phi coefficient								1	2	2		30	17			52
Unused materials	21	6	15	13	5	23	4	4	2	3	2	7	4	10	3	122
Total	201	167	146	111	52	184	71	39	10	11	4	42	28	14	4	1084

Table 2 Example of TSCS table of students' page views generated 2-minutes intervals

The numb	Column label															
Row label	13:02	13:04	13:06	13:08	13:10	13:12	13:14	13:16	13:18	13:20	13:22	13:24	13:26	13:28	13:30	Total
Students	9	7					2	1				1				20
Students	4	1	1	4	1	10	4	1				1				27
Students	8	4				3	2					1				18
Students	4	3	5			3	1	1				1				18
Students	5		2	2	4		1	1								15
Students	6	1	1	3		2	1	1	1				1			17
Students	6	4	6			2	1	1					1			21
Students			8	6			1	1				1				17
Students	4	6	2	3	4	4	4	1		1			1			30
Students	6	3	1	6		5	1	1				1				24
Students		5	2	4		1	3	1				1	1			18
Students		4	3	2	11	1	1	1				1				24
Students		2	4	5	5	2	2	1				1				22
Students		5	2	7		10	1	1				1				27
Students	5	6					2	1				1				15
Students	5	2	8			8						3	1			27
Students	6	5				2	1	2				2				18
Students	6	2		3		3							5			19
Students		5	16	17		8	2	2				2				52
Students	5	6	2			7	3	2				1				26
Students	6	6	1		1	3	1	1					1			20
Students					11	2	1						3	2	2	21
					... More students' records cut here ...											

4. DISCUSSION AND CONCLUSION

Processing of the macro is completed in several seconds. About using the macro during class, the application of the macro to produce a TSCS table for an actual class reveals that such table can be generated without any problems. Depending on the manner by which teacher proceeds with a lesson, however, certain cases have not enough time to use the macro. If students are asked to perform lesson related tasks, such as computing practice, a teacher can run the macro more than once. Aside from enabling teachers to understand the transitions that underlie students' page views, a TSCS table for course items also provide data on variations in students' levels of concentration (Table 1). In the classroom, the teacher manipulated the computer at the teacher's desk and displayed the course materials on the projector. Therefore, the number of students viewing the course materials is smaller because they were looking at the projector screen while listening to the teacher's instruction; i.e. they received the lesson without opening the course materials on their own PC.

Note that certain risks are associated with the use of the TSCS tables. The TSCS table has undeniable possibility of looking at the downloaded materials. Furthermore, after a teacher provides directions on opening a course material, students spend about 1 to 2 minutes accessing the resource. The aforementioned issues should be considered before teachers advance to the next lesson. TSCS tables reflect the viewing behaviors of students over time as transitioning numerical values. During class, teachers can use the tables to visualize the responses of students to instructions. Additionally, the tables provide information regarding which student access teaching materials without following a teacher's instructions and those who exhibit a delay in opening the materials (Table 2). These learners can be distinguished on the basis of transitioning numerical data.

5. ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number 15K00498.

6. REFERENCES

[1] Dierenfeld, H. and Merceron, A. 2012. Learning Analytics with Excel Pivot Tables. In *Proceedings of the 1st Moodle Research Conference* (MRC2012). Retalis, S. and Dougiamas, M. (Eds), 115-121.

[2] Google Analytics. 2016. http://www.google.com/analytics/

[3] Mazza, R. and Dimitrova, V. 2005. Generation of graphical representations of student tracking data in course management systems. In *Proceedings of the 9th International Conference on Information Visualization*. London, UK, July 6-8, 2005.

[4] Mazza, R. and Milani, C. 2004. Gismo: a graphical interactive student monitoring tool for course management systems. *International Conference on Technology Enhanced Learning*. Milan, 1-8.

[5] Konstantinidis, A. and Grafton, C. 2013. Using Excel Macros to Analyses Moodle Logs. In: *2nd Moodle Research Conference* (MRC2013). (4th and 5th October, 2013, Sousse, Tunisia).

[6] Romero, C., Ventura, S. and Garcia, E. 2008. Data mining in course management systems: Moodle case study and tutorial. *Computers and Education*, 51 (1), 368–384. DOI= http://dx.doi.org/10.1016/j.compedu.2007.05.016

[7] Zhang, H. and Almeroth, K. 2010. Moodog: Tracking Student Activity in Online Course Management Systems. *Journal of Interactive Learning Research*. 21(3), 407-429.

Predicting STEM Achievement with Learning Management System Data: Prediction Modeling and a Test of an Early Warning System

Michelle M. Dominguez
Dept. of Educational Psychology & Higher Education, University of Nevada, Las Vegas
4505 S Maryland Parkway #3003
Las Vegas, NV 89154, USA
+001(702)895-3253
doming90@unlv.nevada.edu

Matthew L. Bernacki
Dept. of Educational Psychology & Higher Education, University of Nevada, Las Vegas
4505 S Maryland Parkway #3003
Las Vegas, NV 89154, USA
+001(702)895-4013
matt.bernacki@unlv.edu

P. Merlin Uesbeck
Dept. of Computer Science, University of Nevada, Las Vegas
4505 S Maryland Parkway #3003
Las Vegas, NV 89154, USA
uesbeck@unlv.nevada.edu

ABSTRACT
Learning management systems log users' behaviors, which can be used to predict achievement in a course. This paper examines the implications of data representations (e.g., dichotomous vs. count vs. principled, per learning theory) and applies forward selection algorithms to predict achievement in a biology course. Accuracy is compared across models. The paper closes with a description of an ongoing experiment that employs the prediction model, tests how multiple versions of an early alert message impact students' access of learning resources, and compares the influence of messaging approaches related to personalization and feedback.

Keywords
Learning Management Systems, Prediction Modeling, Early Warning Systems, STEM learning, Learning Theory

1. INTRODUCTION
In response to issues with student performance, retention, progression, and completion [5], universities and educational software providers are developing "early warning systems" to identify students likely to obtain poor outcomes [3]. This paper explores whether logs of students' use of course content can inform models that predict these students' performance. Further, if models can be developed that rely on only behaviors occurring in the earliest weeks of a semester [1], intervention activities can be initiated in time to help students prevent negative outcomes [2].

Undergraduate students utilize a learning management system (LMS) for multiple functions. Based on design features of LMS resources, patterns of student activity may implicate how to represent data in prediction models [4]. For instance, it is more appropriate to model use of a downloadable file as a dichotomous event that should impact learning if it occurs once (indicating that a student has obtained the file) compared to zero times (indicating the student has not). In contrast, resources designed for repeated use online, such as practice quizzes, are best captured as count data. We examine implications of different representations of LMS resource use on the accuracy of prediction models, examine whether the most accuracy model predicts performance in subsequent samples, and whether the model can provide a basis for alerting students about their potential for poor achievement.

2. METHODS

2.1 Participants
For the development of the prediction model, LMS logs capturing behavioral data were gathered for 326 students of an Anatomy and Physiology course at a large, public university in the U.S. Of those sampled, 73% were female and 36% were from underrepresented minority groups. To examine the application of the prediction model on future students, additional samples of 298 and 349 students were drawn from the subsequent Spring and following Fall semesters. All three semesters employed an identical syllabus, an analogous schedule through the observation period, and a cloned set of LMS-hosted materials.

2.2 Materials
Prediction modeling used machine data extracted from server logs of users' behavior-based activity in the LMS from the first four weeks of the course (i.e., prior to any exam). Early warning could then be generated and sent in time for learners to adjust tactics or seek help prior to their first unit exam (i.e., in Week 5). The logs were aggregated and enriched using Splunk [7], a platform for search and modeling of machine data, and tables of metadata about content items. Classification of items into resource types was handled by human research programmers. Models were built and evaluated in RapidMiner [6].

2.3 Procedure
The course that provided data was a traditional large lecture class with a companion site on the LMS, Blackboard Learn. Students could access course materials at any time from the start of the semester, and all use was optional. The frequency and timing of each resource access was recorded and coded by a unique item identifier and time stamp. To represent planful, timely, and recurring use of content items, counts of accesses were captured on a weekly basis. Total use was captured per week and for the four-week period. Behavioral data were merged with performance data. The final grade served as the outcome label. Grades were converted to a binary outcome reflecting students' success (1) or failure (0) to earn a grade of 80%, the minimum "B" score needed to earn credit for STEM majors. Data were parsed into tabular form, enriched, and pivoted into counts per week per student in Splunk. Forward Selection, Weka logistic regression algorithms employing Leave-One-Out cross validation were produced for the models, which were evaluated for accuracy (e.g., κ, recall).

2.4 Model Estimation and Application
Four versions of the data were generated. The first version included the *count* of times a student accessed each content item. The second version treated all data as *dichotomously* used or not used in a period. The third version included *both* count of logs and the dichotomous versions of the data. The final version was a *principled* model guided by learning theory and awareness of instructional design intentions of the instructor; a dichotomous

representation was used for items that could be used only once (i.e. the download of a notes document) and count representations for resources that should provide benefits when used repeatedly (e.g., accessing a quiz to repeatedly self-test).

Based on the Kappa (κ) statistic and supplemented with recall metric (i.e., critical for identifying those predicted to struggle), the most accurate model produced during the test phase was then applied to the subsequent two semesters of the same biology course. Content names and date ranges of access were aligned and all potential attributes, as both dichotomous and count, were transformed using the prediction model equation to calculate *z*-values for all students, which was then converted to probability. A probability greater than 0.5 corresponded to passing with a B or better and a probability less than 0.5 corresponded to C or worse.

3. RESULTS & DISCUSSION

Differences in prediction accuracy appear in Table 1. Representing the data as only count or dichotomous produced models with accuracy better than chance, (κ = .161 and κ = .165, respectively). The model with data as both count and dichotomous improved the accuracy to κ = .224, however the recall of students to be targeted by the early warning system (i.e., those who fail to obtain a B or Better) fell. Compared to the metrics obtained by the first three models, the model employing principled representation produced the best combined accuracy, κ = .212; recall = 84.24%. It appears that drawing inferences from LMS design features and learning theory to make data representation choices maximizes the predictive accuracy of a model. We next tested its subsequent utility for identifying students at risk of poor outcomes.

3.1 Application of Prediction Models to Subsequent Samples

Using the most accurate model (Principled, Table 1), attributes and weights were applied to the new data sets to generate predictions. Kappa decreased to .071 compared to training and testing phase (κ = .212). Recall achieved with spring data was 85.14%, on par with recall obtained with the training (84.24%). This model accurately identified more than 4 of 5 future biology students who would eventually fail to earn a B. Of those labeled, half did obtain a B or Better (precision = 51.85%, initial principled model precision was 63.01%). This level of accuracy is sufficient to warrant consideration of the model for utilization in an early warning system as it is high enough to provide accurate warnings to students at risk of a poor outcome.

4. ONGOING RESEARCH

4.1 Implementation of Early Warning Systems

A follow-up study is currently underway to examine the application of the prediction model in an early alert system and whether issuing an alert to students could change student behavior or achievement. The principled version of the data model was programmed into Splunk in order to calculate the likelihood the students (N = 430) in the current semester would obtain a B or better. An early warning message was sent from the instructor through the LMS correspondence tool. Each message included a salutation, indication of the upcoming exam, and a redirect of the student to helpful resources available on the LMS for students to use (i.e., advice from A or B-earners from prior semesters, about tactics used; modules training students to apply these tactics). The students were randomly assigned to 8 groups, which included varying combinations of the message to test the importance of personalizing the message and framing with feedback. The message was sent Monday of Week 5, four days before their exam.

4.2 Preliminary Findings

Of the 326 students that were messaged, 26.4% accessed the Advice page within 24 hours after receiving the message. In total, 37.4% of the messaged students accessed the Advice page before the exam later that week. Effects on motivation, behavior, and achievement will be analyzed when available.

Table 1. Prediction models using different versions of data and using best model on subsequent semesters

Data representation	κ	Accuracy (%)	Precision (%)	Recall (%)	True: Predicted			
					1:1	1:0	0:1	0:0
count	.16	61	61	82	48	94	34	150
dichotomous	.17	60	63	72	63	79	52	132
both	.22	63	65	73	69	73	49	135
principled	.21	63	63	84	51	91	29	155
Future Semesters								
Spring	.07	53	52	85	33	117	22	126
Fall	.15	58	57	81	56	112	34	147

Note. The baseline for test data versions (count, dichotomous, both & principled) is 56%. The baseline for the Spring use data is 51% and the baseline for Fall use data is 52%.

5. ACKNOWLEDGMENTS

This project was supported by National Science Foundation Award number DRL-1420491, university sponsorship and UNLV Information Technology.

6. REFERENCES

[1] Baker, R., Lindrum, D., Lindrum, M.J., Perkowski, D. (2015) Analyzing Early At-Risk Factors in Higher Education e-Learning Courses. Proceedings of the 8th International Conference on Educational Data Mining, 150-155

[2] Hernandez, P. R., Schultz, P., Estrada, M., Woodcock, A., & Chance, R. C. (2013). Sustaining optimal motivation: A longitudinal analysis of interventions to broaden participation of underrepresented students in STEM. Journal of educational psychology, 105(1), 89.

[3] Jayaprakash, S. M., Moody, E. W., Lauría, E. J., Regan, J. R., & Baron, J. D. (2014). Early alert of academically at-risk students: An open source analytics initiative. Journal of Learning Analytics, 1(1), 6-47.

[4] Macfadyen, L. P., & Dawson, S. (2010). Mining LMS data to develop an "early warning system" for educators: A proof of concept. Computers & Education, 54(2), 588-599.

[5] Nandeshwar, A., Menzies, T., & Nelson, A. (2011). Learning patterns of university student retention. Expert Systems with Applications, 38(12), 14984-14996.

[6] Rapidminer [Computer software]. (2015). Retrieved from http://www.rapidminer.com

[7] Splunk [Computer software]. (2015). Retrieved from http://www.splunk.com.

Comparison of Selection Criteria for Multi-Feature Hierarchical Activity Mining in Open Ended Learning Environments

Yi Dong
Institute for Software Integrated Systems
Vanderbilt University
1025 16th Ave S, Nashville, TN 37212
yi.dong@vanderbilt.edu

John S. Kinnebrew
BRIDJ
283 Newbury St, Boston, MA 02115
john.kinnebrew@gmail.com

Gautam Biswas
Institute for Software Integrated Systems
Vanderbilt University
1025 16th Ave S, Nashville, TN 37212
gautam.biswas@vanderbilt.edu

ABSTRACT

This paper extends our previous work on a Multi-Feature Hierarchical Sequential PAttern Mining (MFH-SPAM) algorithm for deriving students' behavior patterns from their activity logs in an Open-Ended Learning Environment(OELE). The new algorithm is computationally efficient, and we compare the results generated by the two algorithms.

1. INTRODUCTION

Open-Ended Learning Environments [2, 5] present students with a challenging problem-solving task, along with information resources and tools for solving the task. The complexity of the learning task drives a need for dynamic and adaptive scaffolding to help novice students become effective learners. Learner models and formative assessments need to include representations that capture students' problem-solving processes in addition to their knowledge and performance in the task domain. The wealth of data that can be collected from computer-based environments provides opportunities for developing algorithms to accurately model, understand, assess students' learning behaviors and strategies.

In past work, we have developed a hierarchical sequence mining methods [3] for assessing and comparing students' learning strategies and behaviors from their interaction traces collected from OELEs. We then applied a classifier wrapper method [4] to discover smaller subsets of mined patterns that better differentiate students behavior patterns between two groups of students [7]. To address the computational complexity problems with this method while retaining the advantages of the hierarchical approach, this paper applies another selection criteria: Information Gain [6] to derive differential patterns. We conduct experimental studies to analyze student behaviors and compare the two methods.

2. BACKGROUND: MFH-SPAM

Sequential PAttern Mining (Sequential PAttern Mining) algorithm performs a Depth First Search (DFS) traversal to find all possible patterns that exceed a pre-defined frequency threshold from a data set that contains sequences of item sets [1]. SPAM employs a bitmap representation for the patterns and data sequences, which makes it easy to (1) derive pattern extensions and (2) find pattern matches in data sequences during traversal. The DFS search proceeds by extending action sequences with (1)*Sequence-extension* step (*S-step*), which extends pattern by adding a new itemset to the **end** of current pattern sequence, and (2)*itemset-extension* step (*I-step*), which adds a new item to the **last** itemset of a current sequence as an extension.

The MFH-SPAM algorithm further extends the original SPAM algorithm by adding two steps: (1) the *hierarchical-extension* step (*H-step*), which provides a way to get into more details for given actions by bringing in hierarchical representations, and (2) the *feature-extension* step (*F-step*) which makes patterns more informative by associating features with corresponding actions [7]. As a result, MFH-SPAM finds many more patterns compared to the SPAM algorithm. MFH-SPAM also allows for gaps between items(actions) that make up a pattern [3] to accommodate noise tolerance in the action sequences.

In general, even for reasonably-sized domains, the basic MFH-SPAM algorithm returns thousands of patterns, and this presents challenges in extracting the more important patterns that best characterize and differentiate student behavior. Given the computational complexities of the classifier-wrapper method used earlier [7], this paper develops a new selection criterion based on information gain [6] to identify activity patterns that distinguishes students based on their pre- to post-test learning gains measured outside of the system. The information gain for a given pattern P_1 is computed from the reduction in *Shannon entropy* when P_1 becomes known, where *Shannon entropy* for a sample data is a measure of its homogeneity. We focus on analyzing patterns with high information gain that are good differentiators between student groups.

3. CASE STUDY AND RESULTS

We run our case study on a dataset that was generated from an experiment we ran with 98 middle school students who used a learning by teaching environment, *Betty's Brain*, in a science class for a period of approximately six weeks. Learner are tasked to construct a correct causal map of a science process by reading resources, and use the knowledge learned to construct and assess the correctness of their causal map during the study. In one of our current study, students worked on a thermoregulation unit.

The students' learning gains from pre- to post-test provided us with two equally distributed groups: 49 high performers in Group 1, and 49 low performers in Group 2. We then ran the two versions of the MFH-SPAM algorithm: (1) with the classifier wrapper method, and (2) with the information gain methods to select the top 10 patterns that best differentiate the two groups. The results, presented in Tables 1 and 2 respectively, list the mean frequency of usage and the standard deviation for each selected pattern.

Table 1: Classifier Wrapper method.

Pattern	Mean(STD)	Mean(STD)
	High Group	Low Group
editlink;quiztaken	25.9(21.9)	10.6(13.3)
editmap-eff-sup	24.1(17.6)	12.3(11.5)
quiz;editmap	14.0(18.5)	7.5(12.7)
editmap-eff;quiz;expl	11.2(9.7)	5.9(8.6)
quiz;editlink;read	6.1(7.6)	2.3(2.5)
read-shrt;read;editmap;linkadd	4.0(3.3)	2.4(1.7)
read-long	19.8(30.2)	34.0(29.2)
read-shrt;editlink	13.8(9.1)	19.7(10.1)
editmap;quizview	6.8(5.6)	9.7(10.9)
editmap-ineff-unsup;read	5.6(5.1)	8.5(6.4)

Table 2: Patterns with High Information Gain

Pattern	Mean(STD)	Mean(STD)
	High Group	Low Group
quiz	95.3(51.2)	72.9(51.1)
expl	90.4(75.8)	70.0(68.9)
editlink;quiztaken	25.9(21.9)	10.6(13.3)
editmap-eff-sup	24.1(17.6)	12.3(11.5)
editmap-ineff;quiz	20.3(16.3)	14.6(12.7)
editlink;quiz;editmap	16.7(21.4)	7.2(16.1)
quiz;editmap;read	6.4(7.7)	2.8(2.9)
quiz;editlink;read	6.1(7.6)	2.3(2.5)
read-long	19.8(30.2)	34.0(29.2)
take-notes	9.5(11.3)	23.9(24.4)

Both methods find patterns that are good differentiators between the two groups of students. For example, *read-long* (sufficiently long duration read actions) has a high to low performer use ratio of 1 : 2. On the other hand, the quiz followed by an edit link followed by a resource read (*quiz;editlink;read*) has a high to low performer use ratio of 2.75 : 1. Another pattern *editlink;quiztaken* (high to low performer use ratio of 2.5 : 1) found by both methods indicates high performers are better able to use the quizzes(*quiztaken*) to check the correctness of their maps, and to direct their information seeking activities. The classifier wrapper method applying cross validation where decision tree is built multiple times for each chosen pattern, results in larger amount of calculations for information gain, whereas our new method which theoretically finds patterns with highest information gain based on i-frequency, calculates information gain only once for each pattern. Moreover, the new method tends to find shorter patterns because that shorter patterns occupying fewer bits in action sequences for i-frequency based information gain calculation, have lower value of pattern entropy which lead to higher information gain compare to longer patterns with similar usage ratio [6].

4. DISCUSSION AND CONCLUSIONS

In this paper, we have extended an initial version of MFH-SPAM by developing additional selection criteria for pattern selection and also allowing for gaps in the pattern generation from action sequences. The new method is computationally efficient than the previous approach(running time reduced from 28 seconds to 16 seconds) while retaining the strength of finding frequent patterns that are good differentiators.

In future work, we will perform more systematic analysis of the differences between groups using hypothesis testing methods. In addition, we will use correlational analysis to study in more depth the relations between behaviors and performance. We will also work toward using the patterns derived to detect learner behaviors online, and develop scaffolding and hint mechanisms that combine behavior and performance analysis to help students become better learners in OELEs.

5. ACKNOWLEDGMENTS

This work is supported by NSF IIS grant number # 1548499.

6. REFERENCES

[1] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 429–435. ACM, 2002.

[2] G. Clarebout, J. Elen, W. L. Johnson, and E. Shaw. Animated pedagogical agents: An opportunity to be grasped? *Journal of Educational multimedia and hypermedia*, 11(3):267–286, 2002.

[3] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *Journal of Educational Data Mining*, 5(1), 2013.

[4] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, Dec. 1997.

[5] S. Land. Cognitive requirements for learning with open-ended learning environments. *Educational Technology Research and Development*, 48(3):61–78, 2000.

[6] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, Mar. 1986.

[7] C. Ye, J. S. Kinnebrew, J. R. Segedy, and G. Biswas. Learning behavior characterization with multi-feature, hierarchical activity sequences. In *8th International Conference on Educational Data Mining*, June 2005.

A Data-Driven Framework of Modeling Skill Combinations for Deeper Knowledge Tracing

Yun Huang
Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA, USA
yuh43@pitt.edu

Julio D. Guerra-Hollstein
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA, USA
jdg60@pitt.edu

Peter Brusilovsky
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA, USA
peterb@pitt.edu

ABSTRACT

This paper explores the problem of modeling student knowledge in complex learning activities where multiple skills are required at the same time and combinations of skills might carry extra specific knowledge. We argue that in such cases mastery should be asserted only when a student can fluently apply skills in combination with other skills. We propose a data-driven framework to model skill combinations for tracing students' deeper knowledge, and also propose a novel evaluation framework which primarily focuses on the mastery inference quality. Our experiments on two real-world datasets show that proposed model significantly increases mastery inference accuracy and more reasonably distributes students' efforts comparing with traditional Knowledge Tracing models and its non-hierarchical counterparts.

Keywords

complex skill, multiple skill, composition effect, robust learning, deep learning, Knowledge Tracing, Bayesian Network

1. INTRODUCTION

Knowledge Tracing (KT) [2] has been established as an efficient approach to model student skill acquisition in intelligent tutoring systems. The essence of this approach is to decompose overall domain knowledge into elementary skills and map each step's performance to the knowledge level of a single skill. However, KT assumes skill independence in problems that involve multiple skills, and it is not always clear how to decompose overall domain knowledge. Recent research demonstrated that there is additional knowledge related to specific skill combinations; in other words, the knowledge about a set of skills is greater than the "sum" of the knowledge of individual skills [6], some skill must be integrated (or connected) with other skills to produce behavior [9]. For example, students were found to be significantly worse at translating two-step algebra story problems into expressions (e.g., 800-40x) than they were at translating two closely matched one-step problems (with answers 800-y and 40x) [6]. In particular, research on computer science education has long argued that knowledge of a programming language cannot be reduced to a sum of knowledge about different constructs since there are many stable combinations (patterns, schemas, or plans) that have to be taught. We present a data-driven framework for modeling skill combinations and evaluating student models for adaptive tutoring in order to achieve deeper knowledge tracing.

2. PROPOSED FRAMEWORK

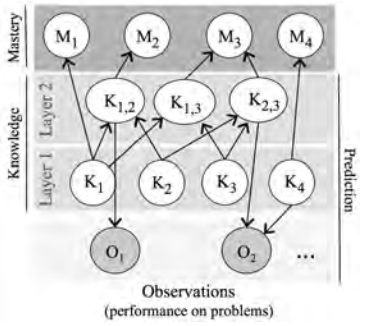


Figure 1: The Bayesian network structure of CKM-HSC.

We construct a Bayesian network called *conjunctive knowledge modeling with hierarchical skill combinations (CKM-HSC)* with the following knowledge structure:

I The first layer consists of basic individual skills (e.g., K_1) that capture the basic understanding of each skill.

II The intermediate layers consist of skill combinations (e.g., $K_{1,2}$), which can be derived from smaller skill units that capture a deeper knowledge level of each individual skill. Now, we consider only skill combinations from two basic individual skills.

III The last layer consists of *Mastery* nodes (e.g., M_1) for each individual skill, which reflects the idea of granting a skill's mastery based on relevant skill combinations' knowledge levels. Now, we compute the joint probability of each relevant skill combination being known as the probability of the current skill being mastered.

To learn the network structure, we propose a greedy search algorithm where a pre-ordering of the skill combination candidates is given as input, and during each iteration, the data likelihood of the network incorporating a new skill combination is compared to that of the optimal network so far. We now replace the search procedure with an empirical thresholding method, which generates an almost identical network with much less time. It selects combinations based on the following criteria: 1) the difficulty difference between the combined skill and its hardest individual one should be positive and large; 2) the difficulty of the combined skills should be high; 3) an item with higher difficulty should be more likely to require combined skills; and 4) each item can only have a limited number of skill combinations. To perform a dynamic knowledge estimation, we use the roll-up mechanism, as in [1]. For performance prediction, we apply Noisy-and gates on item nodes (e.g., O_1) as in [1, 3].

Table 1: Dataset descriptive statistics.

Dataset	#obs.	#items	#skills	avg #skills/item	#users	%correct
SQL	17,197	45	34	5 (from 1 to 10)	366	58%
Java	25,988	45	56	5 (from 1 to 11)	347	67%

To address the limitation of predictive performance metrics [7, 5], we propose a multifaceted data-driven evaluation framework that includes mastery accuracy and effort, the item discriminative index [3], and performance prediction metrics. The basic idea of the mastery accuracy metric is that once a student model asserts mastery for an item's required skills, the student should be unlikely to fail the current item. Meanwhile, the mastery effort metric empirically quantifies the number of practices that are needed to reach a level of mastery for a given set of skills. These metrics extend our recent learner effort-outcome paradigm [5] and Polygon multifaceted evaluation framework [7].

3. STUDIES

We used datasets collected from SQL and Java programming learning systems from 2013 to 2015 at the University of Pittsburgh. Table 1 shows the descriptive statistics (with multiple attempts). We conducted a 10-fold student stratified cross-validation. For each metric, we reported the average value across 10 folds and with a 95% confidence interval, based on the t-distribution. We used the Bayes Net Toolbox to construct all the models. On average, we extracted 14 and 30 skill combinations on SQL and Java datasets.

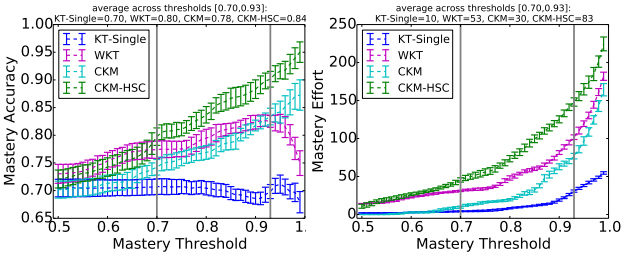


Figure 2: Mastery accuracy and effort comparison on Java dataset. Grey lines denote regions with enough data points to compute mastery metrics and with high enough values to be considered as proper mastery thresholds.

Our first study investigates whether the proposed skill combination incorporated model is better than traditional KT models. We compare classic Knowledge Tracing (KT-Single) [2], Weakest Knowledge Tracing (WKT) [4], and our proposed conjunctive knowledge modeling without (CKM) or with skill combinations (CKM-HSC) (Figure 2). On both datasets, CKM-HSC has a comparable predictive performance to other models, but it has significantly better mastery accuracy than other models. Although it requires more efforts to reach mastery, we think that such "extra" practices is necessary for reaching an acceptable mastery inference accuracy. We further conduct a drill-down analysis for mastery effort by splitting skills into two groups based on whether they involve skill combinations. We find out that for skills that involve skill combinations, WKT would blindly distribute students' efforts among different application contexts, risk students reaching mastery by practicing simple problems, and also guide students to spending more efforts on skills without combinations. On the other hand, CKM-HSC saves students' efforts on basic individual skill understanding and on skills without skill combinations. It requires students to focus more on applying skills in different contexts combined with other skills. We further conduct two studies demonstrating that using a hierarchical structure is better than using a flat independent structure for incorporating skill combinations, and that our modeling can be improved by adding external knowledge (such as expert knowledge or skill combinations' textual proximity) for skill combination extraction. Details are reported in [8].

4. CONCLUSIONS

Our work serves as a first attempt to consider the skill application context for modeling deeper knowledge in a student model using data-driven techniques. We also propose a novel data-driven evaluation framework for such complex skill student models. We only consider pairwise skill combinations as the skill application context; it will be to interesting to consider more complex skill combinations. Such combinations should have a natural connection with the concept of *chunk* in cognitive psychology for defining expertise. Meanwhile, to address the problem of computational complexity we now employ some heuristics. We should explore alternative approaches and more efficient techniques. We will also consider working with larger datasets and datasets with more sparse connections among variables. We expect that our model can provide more benefits when deployed in real-world tutoring systems. For example, it might enable better remediation and raise students' awareness of pursuing true mastery.

5. REFERENCES

[1] C. Conati, A. Gertner, and K. Vanlehn. Using bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, 12(4):371–417, 2002.

[2] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1995.

[3] J. De La Torre. An empirically based method of q-matrix validation for the dina model: Development and applications. *Journal of educational measurement*, 45(4):343–362, 2008.

[4] Y. Gong, J. E. Beck, and N. T. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *Proc. 10th Int. Conf. Intelligent Tutoring Systems*, pages 35–44. Springer, 2010.

[5] J. P. González-Brenes and Y. Huang. Your model is predictive but is it useful? theoretical and empirical considerations of a new paradigm for adaptive tutoring evaluation. In *Proc. 8th Intl. Conf. Educational Data Mining*, pages 187–194, 2015.

[6] N. T. Heffernan and K. R. Koedinger. The composition effect in symbolizing: The role of symbol production vs. text comprehension. In *Proc. 19th Annual Conf. Cognitive Science Society*, pages 307–312.

[7] Y. Huang, J. P. González-Brenes, R. Kumar, and P. Brusilovsky. A framework for multifaceted evaluation of student models. In *Proc. 8th Int. Conf. Educational Data Mining*, pages 203–210, 2015.

[8] Y. Huang, J. Guerra, and P. Brusilovsky. Modeling skill combination patterns for deeper knowledge tracing. In *6th Int. Workshop on Personalization Approaches in Learning Environments (In Submission)*, 2016.

[9] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5):757–798, 2012.

Generating Semantic Concept Map for MOOCs

Zhuoxuan Jiang[1], Peng Li[1], Yan Zhang[2], Xiaoming Li[1]
School of Electronics Engineering and Computer Science
Peking University, Beijing, China
[1]{jzhx,lipengcomeon,lxm}@pku.edu.cn, [2]zhy@cis.pku.edu.cn

ABSTRACT

The task of re-organizing the teaching materials to generate concept maps for MOOCs is significant to improve the experience of learning process, e.g. adaptive learning. This paper introduces a novel and tailored Semantic Concept Map (SCM), and we design a two-phase approach based on machine learning methods to generate it.

1. INTRODUCTION

With the increasing development of Massive Open Online Courses (MOOCs) in recent years, it is believed that how to efficiently re-organize the course materials to serve for better learning is worthy of discussion [6].

In the traditional computer-assisted education, concept map is useful but usually involves domain experts. Considering the large amount of MOOCs, an information system that behaves like an expert and provides the skeleton of a concept map can be more effective.

Unlike partially organized e-textbooks, we can not directly identify concepts from various MOOC materials merely through stylistic features, so machine learning based method is leveraged. Moreover, in order to reduce the cost of labelling, semi-supervised framework is adopted in this paper. Rather than generating various relationships between concepts, we define a novel Semantic Concept Map (SCM) which considers semantic similarity as the only relationship without regard to complex and hierarchy ones. Due to its concision and universality, this map can be applied widely to more courses. Figure 1 shows the two-phase approach including 1) concept extraction and 2) relationship establishment.

2. RELATED WORK

Plenty of work about automatically constructing concept maps has been studied with data mining techniques, such as association-rule mining, text mining and specific algorithms [7]. However, these methods are designed for either specific data sources or special learning settings. Due to the diversity of MOOCs settings, they can hardly be leveraged here.

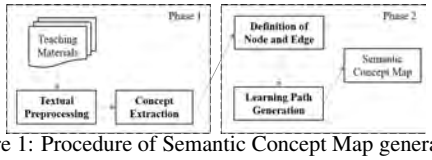


Figure 1: Procedure of Semantic Concept Map generation.

The task of terminology extraction in computer science field is similar to our machine learning based concept extraction [1], but those methods mainly concern about proper nouns or named entity recognition (NER) for generating knowledge graph [5]. Actually this kind of task is corpus-dependent.

3. GENERATING SEMANTIC CONCEPT MAP

Semantic Concept Map. SCM is composed of entities and edges. Formally, denote $SCM = \{C, R\}$ where $C = \{c_1, c_2, ..., c_n\}$ is a set of concepts. Each concept c_i is denoted by a terminology (including phrase), and unique in C. $R = \{r_{11}, r_{12}, ..., r_{ij}, ..., r_{nn}\}$ is a set of relationships between concepts. Each weight value r_{ij} means the degree of semantic similarity between c_i and c_j. The key steps shown in Figure 1 are following.

1.Textual Preprocessing. This step includes tokenization, filtering stop words and removing code and html tags, as well as word segment for Chinese if necessary. We also conduct conflation. All data are randomly shuffled before being learnt and tested, which is partially equivalent to cross-fold validation.

2.Concept Extraction. We leverage CRF+semi-supervised framework to solve this task as a problem of sequence annotation [2]. The labels needed to be predicted of each word are defined as three categories: B, I and O, which respectively mean the beginning word of a concept, the internal word of a concept and not a concept. Feature definition is a key part of machine learning method. Then we design the course- and instructor-agnostic features to meet the diverse materials including stylistic, structural, contextual, semantic and dictionary features. In order to reduce the heavy cost of human labeling, the idea of self-training is leveraged when training data [3].

3.Definition of Node and Edge. The weights of nodes could have different definitions. For example, the more frequent a concept is present in the lecture notes, the more fundamental it is. So the metric of term frequency (tf) can be defined as the node weights, named for *fundamentality*. The diverse teaching materials put together are partitioned to documents corresponding to each video. Moreover, low-frequency concepts may be the key ones of each corresponding unit. So we can define the second metric, Term Frequency and

Table 1: Performance of different concept extraction methods.

	Precision	*Recall*	*F1*
TF@500	0.402	0.500	0.446
TF@1000	0.600	0.746	0.665
BT	0.099	0.627	0.171
SC-CRF	**0.890**	**0.842**	**0.865**
SSC-CRF	0.875	0.783	0.826

Inverted Document Frequency (*tfidf*) which is ideal for quantifying the importance of a concept. As to the weights of edges, the Cosine distance of two word vectors of concepts are defined as the semantic similarity, because the word vectors learnt by word2vec have a natural trait that semantically similar vectors are close in the Cosine space and vise verse [4].

4.Learning Path Generation. The learning path depends on the definition of node and edge in the last step. For example in terms of importance, starting from some concept, each time we choose top k most semantically similar concepts and regard the most important one within the top k as the next node of the path. When choosing the subset of top k candidates, we also consider their locational order of first appearance in the lecture notes.

4. EXPERIMENTS

We collect the teaching materials of an interdisciplinary course conducted on Coursera, including lecture notes (video transcripts), PPTs, questions. The instructors and two TAs help label the data.

We select several baselines to extract concepts from MOOCs materials for comparison. The preprocessing is identical for baselines.

- **Term Frequency (TF)**: This is a statistic baseline.
- **Bootstraping (BT)**: A rule-based iterative algorithm given several patterns which contain true concepts.
- **Supervised Concept-CRF (SC-CRF)**: A supervised CRF with all features but semi-supervised algorithm.

Table 1 shows the performance between baselines and our approach (SSC-CRF). The results also show the necessity of machine learning based methods. Figure 2 manifests that semi-supervised learning is competitive with supervised learning. But considering only half labor consumed, semi-supervised learning is feasible and necessary.

Based on the definitions of node and edge mentioned before, the two kinds of SCMs generated look like Figure 3. Starting from the most fundamental concept, *Node*, the first five successors on the path are: *Edge* → *Element* → *Set* → *Alternative* → *Vote*, which are from basic concepts to advanced ones. Starting from the most important concept, *PageRank*, the first five successors on the path are: *PageRankAlgorithm* → *SmallWorld* → *Balance* → *NashBalance* → *StructuralBalance*. We can see they are not only important along with the course syllabus, but also semantically similar.

5. CONCLUSION

In this paper we mainly propose an approach to re-organize existing teaching materials to generate a novelly-defined SCM for facilitating the learning process in MOOCs. This work is a promising start for content-based adaptive learning since hierarchical and multiple relationships of a complete concept map can be incrementally replenished, and meanwhile this map can be extended to more courses and domains. Experiments show a good efficacy of the semi-supervised

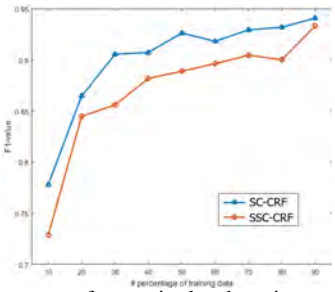
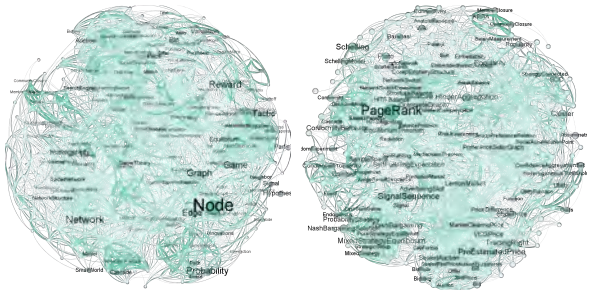


Figure 2: Performance of supervised and semi-supervised learning.



(a) For fundamentality (b) For importance

Figure 3: Two kinds of Semantic Concept Map.

machine learning algorithm and the CRF framework. And the learning paths defined based on SCMs can be humanly modified further to satisfy the requirements of different learners. In future work SCM could be utilized for generating course Wiki via crowdsourcing, hinting concept in forum discussions, etc. Large-scale student knowledge tracing in MOOCs is also doable by associating concepts with questions. Moreover, methods of transfer learning and deep learning may be more effective to extract the abstract concepts from multiple courses and diverse materials.

6. ACKNOWLEDGMENTS

This research is supported by NSFC with Grant No.61532001 and No.61472013, and MOE-RCOE with Grant No.2016ZD201.

7. REFERENCES

[1] K. S. Hasan and V. Ng. Automatic keyphrase extraction: A survey of the state of the art. In *ACL*, pages 1262–1273, 2014.

[2] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML'01*, pages 282–289, 2001.

[3] A. Liu, G. Jun, and J. Ghosh. A self-training approach to cost sensitive uncertainty sampling. *Machine Learning*, 76(2-3):257–270, 2009.

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space, 2013. `http://arxiv.org/abs/1301.3781`.

[5] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich. A review of relational machine learning for knowledge graphs, 2015. `http://arxiv.org/abs/1503.00759v3`.

[6] Z. A. Pardos, Y. Bergner, D. T. Seaton, and D. E. Pritchard. Adapting bayesian knowledge tracing to a massive open online course in edx. In *EDM'13*, pages 137–144, 2013.

[7] C. Romero and S. Ventura. Educational Data Mining: A Review of the State of the Art. *IEEE Trans. on Systems, Man, and Cybernetics*, 40(6):601–618, 2010.

How to Judge Learning on Online Learning: Minimum Learning Judgment System

Jaechoon Jo
Dept. of Computer Science and Engineering
Korea University
Seoul, South Korea
(+82)2-3290-2684
jaechoon@korea.ac.kr

Heuiseok Lim
Dept. of Computer Science and Engineering
Korea University
Seoul, South Korea
(+82)2-3290-2684
limhseok@korea.ac.kr

ABSTRACT
The popularity of online education environment is growing due to the Massive Open Online Course (MOOC) movement. Many types of research in educational data mining (EDM) and Learning Analytics have focused on solving assessment challenges; however, the large number of students enrolled in MOOCs makes it difficult to assess learning outcomes. Thus, it is necessary to develop an automatic learning judgment system. In this study, we designed and developed a minimum learning judgment system that assesses minimal learning using a word game performance measure. In the system, a student watches a video containing educational content and is subsequently tested on information retention by playing a word game that tests the student on the video content. This learning judgment system tests minimal learning of educational content without requiring significant effort from either the instructor or the student. We conducted experiments to show a performance of the system and the result shows about 95% (Pass judgment: 95.1%, Fail judgment: 94.8%) performance.

Keywords
MOOC, Flipped Learning, Judge System, Online Education, Data Collection, Educational Data Mining.

1. INTRODUCTION
Over 10 million people participate in online learning courses, which has resulted in the proliferation of the use of MOOCs. Consequently, the number of online courses that implement online learning platforms, such as Moodle, Coursera, and edX has steadily increased in online education. Online learning platforms provide useful learning data for learner modeling and learning analysis. Learning data provide various types of information that can assess student participation in online courses, such as the number of logins, the number of postings made to discussion boards, and various types of learning outcomes [1]. However, due to the high number of students participating in MOOCs, one critical problem that must be addressed is how instructors can conduct learning assessments that determine learning. Traditional assessment methods are not suitable for online education. Most existing most online learning platforms require a simple quiz and online exam based on traditional assessment methods [2]. Many quizzes and exams can be a burden to both instructors and students. Thus, it is necessary to develop an automatic learning judgment system that can quickly and simply assess learning.

In this paper, we aim to design and develop a minimum learning judgment system. Our approach aims to solve learning assessment challenges in online education in order to minimize the amount of effort required by teachers and learners in assessing learning. Anyone can access and utilize this system [1] at no cost for the purposes of conducting research and collecting educational data. We will present the overall system process and the experiments that were conducted to test the system.

2. MINIMUM LEARNING JUDGMENT
In this paper, we define minimum learning as a behavior state of initial learning, which is automatically determined after a student watches a video and is assessed using a recognition process that measures the frequency effect theory of words used in the video content [3]. In other words, watching video content is the minimal behavior of learning apart from understanding. It does not mean that system can assess understanding of content knowledge.

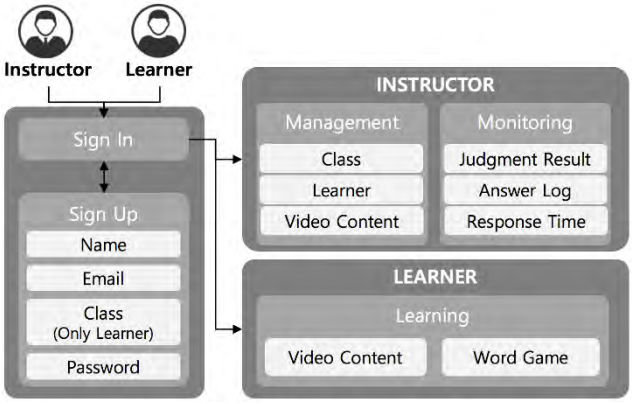


Figure 1. Overall System Process

Figure 1 presents the overall system process for users. After registering an instructor, the instructor can add classes and upload video contents. Words are extracted from the uploaded video content and word frequency is automatically calculated. After registering a learner with a class, the student can learn by viewing video content that the instructor has uploaded. After viewing the video, the student can begin the word game. In the word game, the student decides whether words did or did not appear in the video. The system judges minimum learning by measuring the student's response time and accuracy in the word game. Finally, the

[1] Minimum Learning Judgment System: http://www.mljs.org

instructor checks the minimum learning results, the word game logs and a response time for each word.

The words that appear in the word game use word frequency from uploaded video content and the Sejong corpus (made by www.sejong.or.kr). In order to select words for the word game, words are selected by measuring the weight of each word, which is based on both previous videos that the student learned and on the current video content that student is watching. Each student plays a word game with a different word set in which different weights correspond to different learning logs. The weight of a word is calculated as follow:

$$w_{ij} = tf_{ij} \times log\left(\frac{N}{n}\right) + 1 \quad (1)$$

A weight $w_{ij} > 0$ is associated with each word i in a video content j. Let tf_{ij} refer to the frequency of word i in video content j. Let N refer to the number of video contents viewed by the student in the entire set of video contents. Let n be the number of video contents where w_{ij} appears in N.

In total, 14 words are selected for the word game. The seven highest-frequency words are selected from video content and seven words that have the same word length as the video content words are selected randomly from the Sejong corpus. These latter seven words that do not appear in video content will referred to as "noisy words." The reasoning behind choosing seven words is that the video content is based on short-term memory (STM) [4]. When the word appears, the student chooses the word within two seconds. According to language cognition theory, cognition time of a known word takes between from 700ms to 1200ms [3]. Taking into consideration the conditions that may affect the speed of web environment networks, this system adds and subtracts 500ms to the recorded response time.

3. EXPERIMENTS

3.1 Participants and Analysis

In order to get a criteria score, we conducted an experiment in which we tested 60 undergraduate students. Thirty-two of the students were male, 28 of the students were Female, and the ages of the selected participants ranged from 19 to 27. Each participant viewed video content and then played the word game. Then, participants' attention levels were assessed on a five-point scale using the Likert-type scale. The data collected from the system was analyzed based on the expectation-maximization (EM) algorithm using WEKA. Table 1 presents the results of our analysis.

Table 1. Result of Clustering

Cluster	A	B
Attention	1.004 (SD. 0.027)	3.6084 (SD. 1.0678)
Score	6.0588 (SD. 2.0694)	9.5569 (SD. 2.566)

Cluster A refers to the set of participants who did not pay attention while watching the video content. On average, the members in Group A selected six of the 14 words correctly. Cluster B refers to the set of participants who paid attention while watching the video content. On average, the members in Group B selected 9 of the 14 words correctly. Therefore, the criteria for the minimum learning judgment system correspond to seven correctly selected words.

3.2 Test and Results

Finally, we ran a minimum learning assessment to determine whether learners watched the video content or not. In a test set, 240 undergraduate students participated in the experiment. Participants were divided into two groups: an experiment group, which consisted of 120 students who watched the video content, (Pass) and a control group, which consisted of 120 students who did not watch the video content (Fail). Table 2 presents the results of the test, which measured precision and recall.

Table 2. Result of Test Table

		Real				
		Pass	**Fail**	**Precision**	**Recall**	**F1**
System	**Pass**	118	10	92.1875	98.3333	95.1
	Fail	2	110	98.2142	91.6666	94.8

For the Passing group, the result of minimum learning judgment demonstrated a precision rate of 92% and a recall rate of 98%. For the Failing group, the result of minimum learning judgment demonstrated a precision rate of 98% and a recall rate 91%. Finally, the performance of system shows about 95%.

4. CONCLUSIONS AND FUTURE WORK

This paper presents how a minimum learning judgment system can solve assessment challenges in online education environments by reducing the work required by both instructors and learners. This system shows about 95% performance but it is optimized for the training data set. Thus, we need to conduct further experiments and analyses using machine learning algorithms and educational data mining technologies in order to develop and strengthen our system. Finally, we hope **this system can be utilized by instructors and researchers** for their educational and research purposes.

5. ACKNOWLEDGMENTS

This work was supported by the ICT R&D program of MSIP/IITP. [2016, Development of distribution and diffusion service technology through individual and collective Intelligence to digital contents].

6. REFERENCES

[1] Fazel Keshtkar, Jordan Cowart, and Ben Kingen 2015. Analyzing Students' Interaction Based on their Responses to Determine Learning Outcomes. In *Proceedings of the 8th International Conference on Educational Data Mining, Poster and Demo Papers*, 588-589.

[2] Shumin Jing 2015. Automatic Grading of Short Answers for MOOC via Semi-supervised Document Clustering. In *Proceedings of the 8th International Conference on Educational Data Mining, Poster and Demo Papers*, 554-555.

[3] Jaechoon Jo, and Heuiseok Lim 2015. A Development of Minimum Learning Judgement System of Online Learner, *Korean Association of Computer Education*, 20, 1, 63-66.

[4] Randall, W. Engle 2002. Working Memory Capacity as Executive Attention, *Current Directions in Psychological Science*, 11, 1 (Feb. 2002), 19-23

Guiding Students towards Frequent High-Utility Paths in an Ill-Defined Domain

Igor Jugo Božidar Kovačić Vanja Slavuj
Department of Informatics
University of Rijeka
Croatia
+385 51 584 711
ijugo@inf.uniri.hr bkovacic@inf.uniri.hr vslavuj@inf.uniri.hr

ABSTRACT
This paper presents an exploratory data mining methodology for discovering frequent high-utility learning paths from a database of student interactions with an adaptable tutoring system. The discovered paths are used to present recommendations to students in order to make the learning process more efficient. The novelty of our approach is twofold: a) the process of data preparation, path evaluation and path discovery is completely autonomous; and b) the process is executed on a growing dataset of learning traces while the students are advancing through the knowledge domain. We present the system overview and the obtained results.

Keywords
Sequential pattern mining, computer-based learning environment, high-utility patterns, recommendations.

1. INTRODUCTION
The objective of a tutoring system is to guide each student towards a predefined goal such as completing a lesson, task, or mastering a skill. Guiding students is more complex in ill-defined domains [4] where it is not possible to break down the learning units into single skill tasks, and the students have the freedom to choose/create their own path through the domain. One such web-based system has been developed at our institution to serve as an additional learning platform in a blended learning approach applied in a number of courses. The process presented in this paper is the third and final part (the first two being: 1) a communication layer that enables the system to communicate with DM tools, and 2) a clustering method [3] that discovers groups of students that use the system in a similar manner) of a new infrastructure developed with the goal of improving the adaptivity of our system [2].

While attempting to master the knowledge domain presented in our system, each student creates a large number of learning paths. Most of the students will need multiple interactions with a unit until it is mastered/completed, e.g., after a failed attempt they realize they need to learn some other (lower-level) units and then they come back to complete the first unit. The objective of the system is to offer recommendations to students about which unit to select (when the student is just starting a session or a new learning "run") or which unit to learn next (right after finishing learning a unit). For this, we need to discover productive frequent paths leading to, and following after, each unit. To discriminate between productive and unproductive frequent patterns we decided to construct a new dataset based on the database of learning paths and then feed that dataset into a high-utility sequential pattern mining algorithm USPAN [5] which requires a sequence database that contains both the unit IDs and their "profit" (in our case – the calculated efficiency of each path).

2. DISCOVERING FREQUENT HIGH-UTILITY PATHS
The system supports two types of learning activities:

a) LEARNING - presenting learning materials followed by a question about the unit, and initial questions about the connected underlying units (units below in the domain structure created by teacher). If the student answers all the questions correctly, the path is considered optimal and the change in the student's overlay model is calculated. If the student offers an incorrect answer to a question about a connected unit, the system will transfer the student to learn that unit, and the whole process is recursively repeated. Therefore, one learning "run" can consist of a number of learning units and a number of questions answered;

b) REPETITION - answering a series of questions about a unit without presenting learning materials. A visualization of four possible paths for a sample domain consisting of five units (A-E) is presented in Figure 1.

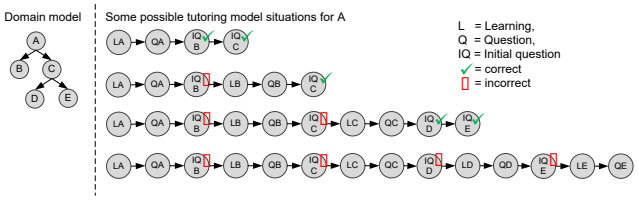


Figure 1. Possible variations in learning path lengths

The basic components for profit calculation, based on four paths presented in Figure 1, are presented in Table 2. Each unit has a set threshold value t that the student has to reach (by answering the questions). The current value of t for each unit in the domain model represents the student's model.

Table 1. Path profit calculation

UNITS									
A	**B**	**C**	**D**	**E**	**∑**	**PL**	**Ls**	**Qs**	**IQs**
t=10	t=10	t=6	t=8	t=10					
+2	+0.2	+0.2			2.4	4	1	1	2
+3	-0.1 +2	+0.2			5.1	6	2	2	2
+2	-0.1 +2	-0.1 +2	+0.2	+0.2	6.2	10	3	3	4
+1	-0.1 +1	-0.1 +1	-0.1 +2	-0.1 +2	6.6	14	5	5	4

The following expression is used to calculate "profit" P of path x:

$$P_x = \left(\frac{\sum_{i=1}^{Units} cq_i/t_i}{Q} + \frac{\sum_{i=1}^{Units} ciq_i/t_i}{IQ} \right) * \frac{PL_{min}}{PL}$$

We summarize the changes *c* that followed from answering a question about each of the units (Units) occurring in *x*, divided by the unit threshold value *t*. This accounts for the difficulty of the presented questions. The sum is then divided by the total number of questions answered (Q). The same is done for initial questions (IQ). Finally, the total change is multiplied by the difference between minimal and actual path length (PL). This penalizes longer paths as they are caused by incorrect answers to initial questions. Minimum path length (PL_{min}) is calculated based on the number of units added to the learning structure at the time the learning activity took place. The tutoring model determines the number of items in the learning structure based on the student's overlay model state, e.g., according to Figure 1, if the student starts the LEARNING activity with unit A, having previously completed units B and C, the tutoring module will not add any units to the learning structure (except for A, making $PL_{min} = 1$).

After the learning traces of all the students that are using the knowledge domain have been evaluated, they can be transformed into a sequence database for the USPAN algorithm. The transformation algorithm creates two databases for each unit in the domain – a set of paths consisting of units learned before the current unit ("prefix") and a set of paths consisting of units learned after ("suffix"). Each transformed sequence has a maximum length of 6 units. Both datasets are then converted to the correct format of the USPAN algorithm implementation in SPMF [1]. The system is now ready to discover high-utility frequent paths (HUFP). We run the algorithm on each dataset under the condition that a unit has been learned by at least five students, i.e., we must have a minimum of five paths in the dataset, although there can be much more if the students have been struggling with the unit. When the process is complete, all the discovered high-utility frequent paths are written to the database. Once the system has updated the HUF paths database the recommendation selection algorithm chooses the unit to be recommended at the beginning and the end of each learning activity. The algorithm considers: a) the student model; b) whether the unit has already been recommended and/or followed by this student; and c) which recommendation was most followed by other students.

3. RESULTS

We tested our system in two different knowledge domains, with 31 and 69 learning units, used by 30 and 20 students, respectively. The results are presented in Table 2. The "D.SET" column contains the number of learning traces in the system at the time the process was executed. The number of HUFPs discovered at each execution (divided by "prefix" and "suffix" paths) is presented in the next three columns. Column "UNITS" presents the number of units for which HUFPs were discovered. As expected, the unique number of units reached the total number of units in the domain in last two executions. The number of recommendations presented to students and the unique number of units for which the recommendations were presented are displayed in the next two columns.

Table 2. Results for first domain

No	D.SET	HUFPs	PRE	SUF	UNITS	REC.	UN. REC	FOL.
1	1206	21	5	16	7	12	5	5
2	1616	35	20	15	15	83	24	39
3	2068	115	65	50	22	204	15	36
4	2504	121	79	42	18	225	15	12
5	2912	89	52	37	21	47	12	13
6	3314	418	227	191	31	417	23	35
7	3604	538	289	249	31	332	24	22

Finally, the last column presents the number of recommendations followed (clicked) by students. The percentage of followed versus total number of recommendations varied from 5 to 47 percent. Further analysis will be performed to evaluate the overall impact of the recommendation mechanism on the learning process.

4. CONCLUSION

The presented methodology was implemented in a web-based ITS and tested on two different domains. We believe that the main improvements to the system can be made in: a) the interaction-to-path transformation algorithm, by implementing additional logic to recognize branch/level changes in the domain hierarchy which can reflect student's strategy, and b) the recommendation selection algorithm, by implementing additional logic to minimize repetition and optimize the selection process.

5. ACKNOWLEDGMENTS

This research is a part of the Project "Enhancing the efficiency of an e-learning system based on data mining", code: 13.13.1.2.02., funded by the University of Rijeka, Croatia.

6. REFERENCES

[1] Fournier-Viger, P. et al. 2014. SPMF: a Java Open-Source Pattern Mining Library. *Journal of Machine Learning Research*, 15, 3389-3393.

[2] Jugo, I., Kovačić B. and Slavuj, V. 2016. Increasing the Adaptivity of an Intelligent Tutoring System with Educational Data Mining: a System Overview, in *Int. Journal of Emerging Technologies in Learning*, 11, 3.

[3] Jugo, I., Kovačić, B. and Tijan, E.. 2015. Cluster analysis of student activity in a web-based intelligent tutoring system, in *Pomorstvo: journal of maritime studies*, 29, 80-88.

[4] Lynch, C. et al. 2006. Defining Ill-Defined Domains; A literature survey. *In ITS2006: Proc. of Intelligent Tutoring Systems Ill-Defined Domains Workshop*, Taiwan, 1-10.

[5] Yin, J., Zheng, Z., & Cao, L. 2012. Uspan: an efficient algorithm for mining high utility sequential patterns. In *18th ACM SIGKDD international conference on Knowledge discovery and data mining,* 660-668.

Computing Pointers Into Instructional Videos

[Extended Abstract] *

Andrew Lamb
Stanford University
andrew.lamb@stanford.edu

Jose Hernandez
Stanford University
josehdz@stanford.edu

Jeffrey Ullman
Stanford University
ullman@cs.stanford.edu

Andreas Paepcke
Stanford University
paepcke@cs.stanford.edu

ABSTRACT

We examine algorithms for creating indexes into ordered series of instructional lecture video transcripts. The goal is for students and industry practitioners to use the indexes towards review or reference. Lecture videos differ from often-examined document collections such as newspaper articles in that the transcript ordering generally reflects pedagogical intent. One challenge is therefore to identify where a concept is *primarily* introduced, and where the resulting index should thus direct students. The typically applied TF-IDF approach gets tricked in this context by artifacts such as worked examples whose associated vocabulary may dominate a lecture, but should not be included in a good index. We contrast the TF-IDF approach with algorithms that consult Wikipedia documents to vouch for term importance. This method helps filter the harmful artifacts. We measure the algorithms against three human-created indexes over the 90 lecture videos of a popular database course. We found that *(i)* humans have low inter-rater reliability, whether they are experts in the field or not, and that *(ii)* one of the examined algorithms approaches the inter-rater reliability with humans.

1. INTRODUCTION

Lecture videos of online classes are clumsy when students wish to review course materials. It is impossible to access just a particular portion of interest. A solution would be an automatically created index similar to the reference at the end of a book. The facility would allow access into portion of videos where a particular topic is discussed.

We compared several algorithms that create such an index for every course video. Raw material are the closed caption files that are often available for educational video. Those files contain transcripts of the audio, paired with timing information at roughly sentence granularity.

We paid three humans with varying domain expertise to carefully index the video transcripts from a Stanford online database course. We compared the three resulting indexes to each other, and to results from the algorithms. We make the three reference indexes and the database course video caption files available to the public in hope of eliciting indexing approaches beyond those that we explored.

*A full version of this paper is available at `http://ilpubs.stanford.edu:8090/1140/1/indexer.pdf`

2. EXPERIMENTS

Our first experiment took a traditional approach, selecting words for the index that appeared disproportionately often in certain lectures (TF-IDF [1]). We then incorporated lexical information, by only considering phrases that followed certain part-of-speech patterns. Finally, we introduced external knowledge from Wikipedia into an algorithm's indexing decisions. Note that none of the algorithms included supervised learning, as we do not assume the existence of a training set for all courses. The following subsections introduce the algorithm (families) beyond the TF-IDF version.

2.1 Leveraging Linguistic Information

The first algorithm tags parts of speech in the lecture transcripts. It then extracts as index candidates phrases that consist of adjectives followed by one or more nouns. For example, "equality condition" or "XML data" would be included.

2.2 Adding External Knowledge

Note that phrases gain importance because of both their role in a document but also from their semantic meaning in the broader world. Variants of our next algorithms therefore integrate Wikipedia as a knowledge source.

2.2.1 Boosting Documents

The first variant concatenates to each lecture a closely related Wikipedia page, and then uses the techniques of Section 2.1 to choose phrases for the index. For example, lecture title "View Modifications Using Triggers", yields as the first Wikipedia result a page titled "Database trigger." This page is appended to the lecture transcript. Using either n-grams or adjective-noun phrases as candidate keywords, the algorithm chooses phrases with TF-IDF over the combined document for the index.

2.2.2 Boosting Phrases

This algorithm first creates a list of candidate index terms using adjective-noun phrases. These candidates are ranked by their TF-IDF score summed over **all** Wikipedia documents.

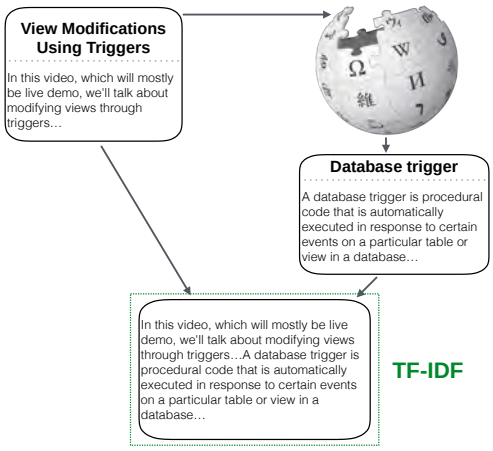


Figure 1: The Document Boosting algorithm searches for a Wikipedia page using the title of the lecture, concatenates the result to the lecture, and then runs TF-IDF over the combined document.

Next, this global candidate ranking is combined with a basic TF-IDF approach to form a final score that combines global knowledge (from Wikipedia) with local knowledge (from the specific lecture video).

We also experimented with only boosting phrases of at least two words, based on the intution that longer phrases are often meaningful, but appear infrequently and are therefore given low scores by TF-IDF. We call this alternative "Phrase Boosting N-Grams" in Figure 3.

Rank	Phrase
1	**view**
2	**materialized view**
3	materialized
4	**query**
5	**view query**
6	**virtual view**
7	**modify**
8	user query
9	**base table**
10	**modify command**
11	**index**
12	insert command
13	multivalued dependency
14	**database design**
15	user

Figure 2: The top 15 keywords from 'Materialized Views' by Phrase Boosting with N-grams. Phrases that also appear in the gold index are marked in bold.

2.3 Results

We evaluated each algorithm by computing Cohen's Kappa agreement between the algorithm and a gold set created by unifying two of the human indexes[1]. We chose a widely employed inter-rater reliability measure because indexing is highly subjective. Given this absence of absolute truth we therefore treated the algorithms as we would have measured reliability of an additional human indexer.

[1]One of the human indexes was excluded because it sometimes included words that did not appear in the lecture.

Kappa values do not have a universally agreed upon interpretation, but values in the range we observe (about 0.15 to 0.3) have been interpreted as indicating "slight" to "fair" agreement. We measured agreement of 0.325 between the humans in the gold index. This value is therefore the measure to beat.

Algorithm	κ
TF-IDF	0.205
TF-IDF with Adjective-Noun Chunks	0.079
Document Boosting	0.209
Document Boosting with Adjective-Noun Chunks	0.142
Phrase Boosting	0.204
Phrase Boosting N-Grams	**0.237**

Figure 3

The metrics for all of the algorithms are shown in Figure 3. The Phrase Boosting N-Grams algorithm, which favors longer words, performed best with a Cohen's Kappa of 0.237. The Document Boosting algorithm is able to slightly improve on TF-IDF, by filtering superfluous keywords using the external knowledge from Wikipedia. Note that Cohen's Kappa can sometimes be problematic when using an unbalanced dataset. In our full paper, we evaluate the algorithms with complementary metrics to guard against potential pathological cases.

Figure 2 shows the set of keywords extracted from a lecture on 'Materialized Views' by the Phrase Boosting with N-grams algorithm, in Figure 2 . Of the top 15 keywords marked by the algorithm, 11 were included in the gold index marked by humans (for this lecture there were 18 keywords in the gold set), and the algorithm produces a ranking that is similar to the humans. Of the keywords ranked highly by the algorithm that were not in the gold index, some ('materialized', 'insert command', 'multivalued dependency') are relevant to the course, but perhaps not essential to the specific lecture. The last two keywords, 'user' and 'user query' expose a weakness of the algorithm, where it is difficult to discern phrases that are used frequently, but not essential to the lecture concept.

3. CONCLUSION

We started to tackle the task of choosing the most important phrases from a collection of lectures, to construct a random-access index analogous to those in the back of books. Going forward we will use this capability to construct student support facilities such as automatically answering learner questions with references to relevant lecture clips, and recommendation tasks, such as finding the best study materials given a student's progress through a course.

4. REFERENCES

[1] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 32 Avenue of the Americas, New York, NY 10013-2473, USA, 2008.

Hierarchical Cluster Analysis Heatmaps and Pattern Analysis: An Approach for Visualizing Learning Management System Interaction Data

Ji Eun Lee
Utah State University
jieun.lee@aggiemail.usu.edu

Mimi Recker
Utah State University
mimi.recker@usu.edu

Alex J. Bowers
Columbia University
bowers@tc.columbia.edu

Min Yuan
University of Utah
min.yuan@eccles.utah.edu

ABSTRACT
This paper presents a form of visual data analytics to help examine and understand how patterns of student activity – automatically recorded as they interact with course materials while using a Learning Management System (LMS) – are related to their learning outcomes. In particular, we apply a data mining and pattern visualization methodology in which usage patterns are clustered using hierarchical cluster analysis (HCA) then visualized using heatmaps to produce what is called a clustergram. We illustrate the application of this methodology by building two clustergrams in order to explore university students' LMS activity patterns using both semester and weekly summary data. The resulting clustergrams reveal differences in LMS usage between high-achieving and low-achieving/dropout students.

Keywords
Hierarchical Cluster Analysis, Heatmap, Learning Management System, Visual Data Analytics

1. INTRODUCTION
With the explosive growth in the use of LMS to support instructional activities, several recent studies have applied Educational Data Mining (EDM) to analyze the vast datasets collected by LMS. Results from such studies can help identify at-risk learners, monitor student performance, and inform course re-design [4, 5].

Some approaches tend to take a variable-centered approach, examining features and trends in key usage variables. In contrast, a person-centered approach can highlight individual sub-groups of students that share common data patterns [1, 7], that when pattern analyzed, link to important differences in overall course or educational outcomes. In this way, data points are not aggregated, thereby obscuring their individual patterns [6].

This study takes the latter, person-centered approach. As a form of visual data analytics, we describe and apply a data mining and pattern visualization methodology, in which usage patterns are clustered using hierarchical cluster analysis (HCA) then visualized using heatmaps to produce what is called a *clustergram* [1]. We illustrate the application of this methodology by analyzing data collected from a widely used LMS, Canvas. In particular, we address two questions: To what extent do clustergrams help understand patterns of student activity in the course? How do these patterns of activity relate to student learning outcomes?

2. BACKGROUND
2.1 EDM and LMS
Much prior EDM research applied to LMS data has typically taken a variable-centered approach by examining usage at an aggregated level [3]. While useful, these results aggregate and average users' behaviors, and thus make it difficult to recognize the diverse patterns displayed by different groups of users [6]. Thus, in the present study, we take a more person-centered approach to visually investigate what sub-groups of students may share common patterns, and how these relate to their learning outcomes.

2.2 Hierarchical Cluster Analysis Heatmaps
HCA is a multivariate statistical method for classifying related units in an analysis across high dimensionality data. More recently, HCA has been combined with heatmap visualizations, called a *clustergram* [1]. The clustergrams represent each participant's row of data across each of the columns of variables as a color block, using stronger intensities of one color to represent lower levels of the variable, and increasing intensities of a different color to represent higher levels. We apply cluster analysis heatmap visualizations to Canvas LMS data from a large, online course. In this way, we test the utility of the analysis and visualization technique when applied to the potentially larger data patterning and visualization issues around these types of student interaction data.

3. METHODS AND DATA SOURCES
The data are drawn from a larger dataset containing all student recorded by the Canvas LMS at a medium-sized U.S. western university. For the present study, we extracted the student interaction data from a large (N=139) introductory level mathematics online course taught during the fall 2014 semester.

Two clustergrams were built, one with semester summary data and the other with weekly summary data. First, for the clustergram using the semester summary data, all student activity data were transformed to z-scores in order to standardize variance. HCA was applied to cluster both rows and columns. Color gradients ranges from colder blue for -3 SD below the mean to a hotter red for value +3 SD above the mean. Second, for the clustergrams using the weekly summary data, we used raw data and HCA was applied to only the rows. In addition, we applied k-means clustering on the rows for more precise interpretation of clustergrams. Lastly, student final course grade was included as an overall outcome variable in the final column. For all analyses, we used the R studio with the "ComplexHeatmap" packages. Regarding algorithms, the clustergrams were clustered using K-means, then HCA (using average linkage and Euclidean distance) was applied to each row-cluster.

4. RESULTS
4.1 Clustergram using semester summary
Figure 1 presents a section of the clustergram using the semester summary data. As shown in Figure 1, most students with lower

activity (Cluster 1) either received a grade of F or withdrew (W) from the course, whereas many students with higher activity (Cluster 3) received a grade of A.

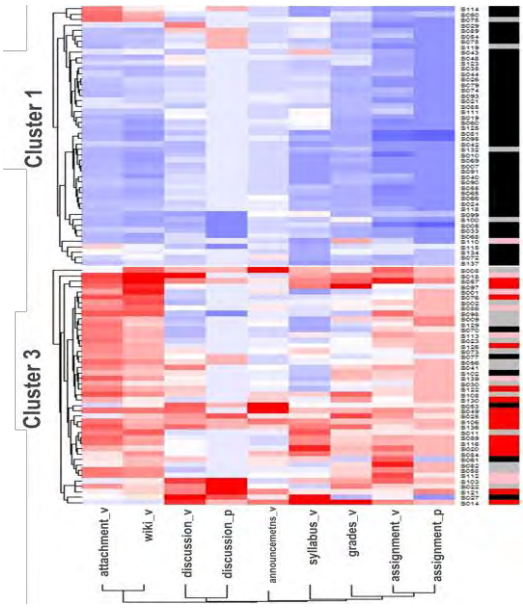


Figure 1. Section of the clustergram using the semester summary data (for full image: goo.gl/Y7VFHJ)

A correlational analysis revealed that the variables related to engaging with 'assignment' features had the highest positive correlations with final grades (r = .70, p < .05). Variables related to views of grades (r = .56, p < .05), wiki (r = .42, p < .05), syllabus (r = .35, p < .05), and attachments (r = .35, p < .05) had the next highest positive correlations with final grades. The remaining variables (views of announcements, participation in discussions) were not significantly correlated with final grades.

4.2 Clustergram using weekly summary

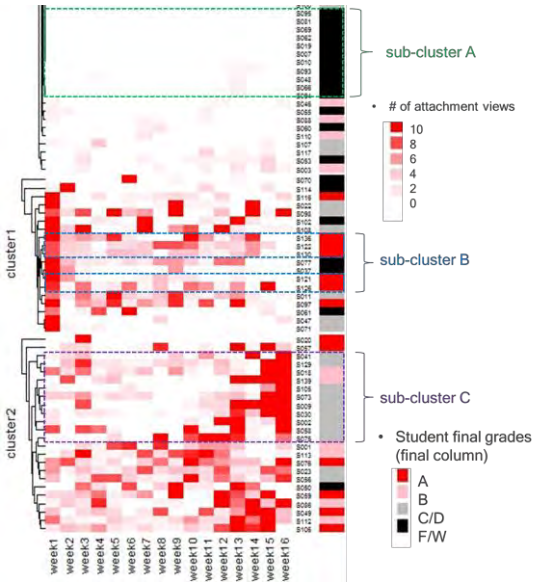


Figure 2. Section of the clustergram for the number of students' attachment views by week (for full image: goo.gl/IwcCch)

In order to investigate how student activities changed over the course of the semester, we built a clustergram using the weekly summary data. Figure 2 presents a section of the clustergram for the number of students' 'attachment' views by week.

The clustergram shows that the students with a grade of A (sub-cluster B) showed relatively consistent views of attachments over the course of the semester. Interestingly, the students with a grade of A (sub-cluster B) tended to show higher attachment views at the beginning of the course and more consistently throughout the semester. However, the students with grades of C/D (sub-cluster C) tended to have higher attachment views at the end of the course, representing perhaps a less-successful 'cramming' strategy.

5. CONCLUSION

This study demonstrates the utility of cluster analysis heatmap visualizations as a means to use visual data analytics to examine student patterns of activity at different grain sizes (week vs. semester). Combining this technique with the large sets of LMS provides a unique opportunity to examine the patterns of student activity as they relate to overall student outcomes. This type of visual data analytics expands the number of tools available for instructors and administrators to help identify the features and specific LMS interaction data that are most useful to their students. As recent critiques of LMS interaction data have shown that past analytic methods are insufficient to understand the rich complexity of how students learn through an LMS [2], this study provides an additional means to approach these complex data analytic issues.

6. REFERENCES

[1] Bowers, A.J. 2010. Analyzing the longitudinal K-12 grading histories of entire cohorts of students: Grades, data driven decision making, dropping out and hierarchical cluster analysis. *Practical Assessment, Research & Evaluation.* 15, 7, 1-18.

[2] Koedinger, K. R., D'Mello, S., McLaughlin, E. A., Pardos, Z. A., and Rosé, C. P. 2015. Data mining and education. *Wiley Interdisciplinary Reviews: Cognitive Science*. 6, 4 (Jul. 2015), 333-353.

[3] Macfadyen, L. P., and Dawson, S. 2010. Mining LMS data to develop an "early warning system" for educators: A proof of concept. *Computers & Education*. 54, 2 (Feb.2010), 588-599.

[4] Picciano, A. G. 2014. Big data and learning analytics in blended learning environments: Benefits and concerns. *International Journal of Artificial Intelligence and Interactive Multimedia*. 2, 7 (Sep. 2014), 35-43.

[5] Romero, C., Ventura, S., and García, E. 2008. Data mining in course management systems: Moodle case study and tutorial. *Computers & Education*. 51, 1 (Aug. 2008), 368-384.

[6] Wilkinson, L., and Friendly, M. 2012. The history of the cluster heat map. *The American Statistician*. 63, *2* (Jan, 2012), 179-184. DOI= http://10.1198/tas.2009.0033

[7] Xu, B., and Recker, M. 2011. Understanding teacher users of a digital library service: A clustering approach. Journal of Educational Data Mining. 3, 1 (Oct. 2011), 1-28.

Understanding Engagement in MOOCs

Qiujie Li
School of Education
University of California, Irvine
Irvine, 92617
qiujiel@uci.edu

Rachel Baker
School of Education
University of California, Irvine
Irvine, 92617
rachelbb@uci.edu

ABSTRACT
Previous studies about engagement in MOOCs has focused primarily on behavioral engagement and less attention has been paid to cognitive engagement. This may lead to incomplete or even incorrect understandings about students experience and learning in MOOCs. In this study, we use number of lectures watched as a proxy for behavioral engagement and number of pauses in lectures watched as a proxy for cognitive engagement. Results show that a large proportion of students who were behaviorally engaged (watching lectures) were not cognitively engaged—they almost never paused the lectures or they paused fewer and fewer times as the course went on. This may indicate that being behaviorally engaged does not necessarily mean being cognitively engaged. In addition, we also found that students' number of pauses in lectures is positively associated with achievement and improves the prediction of achievement.

Keywords
Cognitive engagement, behavioral engagement, MOOCs

1. INTRODUCTION
Engagement in MOOCs is usually measured by whether students complete learning activities or not (e.g. watching lectures and submitting assessments) and low engagement is used as an indicator of "at-risk" students [4]. However, studies of school engagement have proposed that engagement has three components: behavioral engagement, cognitive engagement, and emotional engagement, and that measuring engagement solely as task completion may focus only on behavioral engagement and overlook the multifaceted nature of engagement [1]. To explore the importance of cognitive engagement in MOOCs, this study measured both behavioral engagement and cognitive engagement in MOOC lecture watching to see: 1) whether individuals who were behaviorally engaged were also cognitively engaged, and 2) whether cognitive engagement adds information that is helpful in predicting academic achievement.

1.1 Behavioral engagement
Most of previous studies about engagement in MOOCs have focused on behavioral engagement: participation in academic activities [1]. One of the most commonly used engagement indicators in MOOC studies is participation in lecture watching. For instance, in the most frequently cited paper about engagement patterns in MOOCs, Kizilcec et al (2013) measured student weekly engagement as a function of whether they watched any lecture and submitted any assessment. By using these metrics of task completion, this study inherently conceptualized engagement as behavioral engagement. Similarly, measurements centered around behavioral engagement, such as time spent on lecture resources, have also been used in studies about the relationship between engagement and dropout [4].

1.2 Cognitive engagement
Cognitive engagement refers to the psychological investment in learning and ranges from memorizing to using self-regulated strategies to promote one's understanding [1]. In this study, we measure student's weekly cognitive engagement by how often they paused the lectures they watched (i.e., students stop the lecture while watching it). Some studies about MOOCs have explored the possibility of using video lecture clickstream data, the record of student click events, to measure cognitive engagement [3]. Among all the click events, the pausing event may indicate a higher level of cognitive engagement [3].

2. METHODS

2.1 Sample
This study uses data from one Coursera MOOC, Pre-calculus, offered by University of California, Irvine. It began on October 7th, 2013 and lasted for ten weeks. 50,676 students registered the course and data on 19,548 students who watched at leased one lecture after registration was used in this study.

2.2 Measurement
In this study, weekly behavioral engagement was measured by the number of lectures student watched each week while weekly cognitive engagement was measured by the number of pauses in lectures watched in a given week. In addition, we measured weekly academic achievement in two ways: students' total quiz score (the sum of scores a student got on each quiz he/she attempted each week) and students' average quiz score (the average score on quizzes attempted each week).

2.3 Analysis
We applied a standard clustering technique, K-means, to discover student engagement patterns based on the two measurements to see whether individuals who were behaviorally engaged were also cognitively engaged. We first standardized the engagement score within each week to take into account the difference in participation across weeks and thus to cluster students based on their relative similarity in engagement within each week. Then, we performed the clustering analysis separately for behavioral engagement and cognitive engagement. To get an optimal "goodness of fit" for the data, cluster silhouette, a measure of how similar an individual is to his/her own cluster compared to other clusters, was used to determine the number of clusters. For behavioral engagement, 4 to 9 clusters produced similar cluster

silhouette (above 0.7) and for cognitive engagement, 4 to 8 clusters produced similar cluster silhouette (above 0.6). Accordingly, we performed cluster analysis with all the possible choices. Finally, we chose 4 clusters for both of the two measurements because it gave us enough individuals in each cluster and all the clusters made sense from an educational perspective. In addition, to answer the second research question, we used regression with individual fixed effect to test whether cognitive engagement could predict academic achievement after controlling for behavioral engagement.

3. RESULTS

3.1 Clusters based on different engagement

The four types of behavioral engagement trajectories are: *1)"Strong enders"* (n=157; 0.8%) who watched more lectures than other groups and their average number of lectures watched decreased in the first six weeks but then increased to 50 at the end of the course; *2)"Slow decreasers"* (n=1367; 7.0%) who had a very similar pattern as "stronger enders" except that they kept watching fewer and fewer lectures till the end of the course; *3)"Quick decreasers"*(n=1598; 8.2%) who started at the same place as both "strong enders" and "slow decreasers", but the number decreased at a much faster rate; and *4)"Disengagers"*(n=16426; 84.0%) who watched around 2 lectures in week 1 on average and the number was kept under 1 for the following 9 weeks.

The four types of cognitive engagement trajectories are:*1)"Active stoppers"*(n=41; 0.21%) who, on average, paused each of the lecture they watched more than 10 times in most of the weeks; *2)"Constant stoppers"*(n=367; 1.9%) who, on average, paused each lecture they watched around 5 times in most of the weeks; *3)"Switchers"*(n=1719; 8.8%) who started at the same place as "constant stoppers", but their average number of pauses in lectures watched decreased quickly in the following weeks; and *4)"Continuers"*(n=17421; 89.1%) who almost never paused the lectures they watched or they didn't watch any lectures at all in some of the weeks.

Combining the two types of engagement (see Figure 1), we found that students in clusters with higher levels of behavioral engagement had a larger proportion of individuals who were cognitively engaged. For example, compared with "disengagers" and "quick decreasers", "strong enders" and "slow decreasers" have a smaller percent of "continuers" and larger percent of both "active stoppers" and "constant stoppers". However, being behaviorally engaged does not necessarily mean being cognitively engaged. For example, even though "strong enders" and "slow decreasers" watched the most lectures every week, around 45% them conducted fewer and fewer pauses as the course went on (defined as "switchers") and more than 20% of them almost never paused the lectures (defined as "continuers").

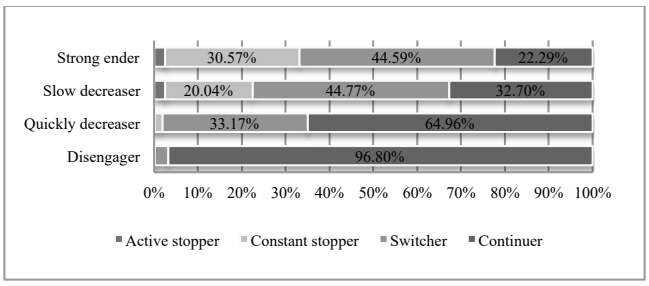


Figure 1. Distribution of cognitive engagement trajectories

3.2 Cognitive engagement and achievement

Using individual fixed effect model (see Table 1), we found that the number of pauses in lectures watched is predictive of both total and average quiz score after controlling for the number of lectures watched. For total quiz score, one more pause is associated with 0.33 points increase in total quiz score and 0.23 points increase in average quiz score. In addition, for both total and average quiz score, the models with the number of pauses in lectures watched fit significantly better than the models that only have number of lectures watched as the predictor. Overall, the results show that our measurement of cognitive engagement is positively associated with achievement and it can make a unique contribution in predicting achievement.

Table 1. Regression of engagement on academic achievement with individual fixed effect

	Total score		Average score	
Number of lectures	0.71^{***}	0.69^{***}	0.04^{***}	0.02^{***}
	(0.004)	(0.005)	(0.001)	(0.001)
Number of pauses per lecture		0.33^{***}		0.23^{***}
		(0.019)		(0.005)
N	79174	79174	79174	79174
R^2	0.281	0.284	0.017	0.054

$^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$

4. DISCUSSION

Our preliminary results indicate that it is important to take into account cognitive engagement. First of all, using only behavioral engagement may lead to an incomplete or even incorrect understanding about the activeness of students. As we found in this study, some students had relatively high behavioral engagement while decreasing or low cognitive engagement. We may fail to identity some "at-risk" students who visited most of materials but didn't truly engage with the content if we only measure behavioral engagement. In addition, cognitive engagement is found to have its unique contribution in predicting academic achievement and thus can give instructors extra information about student performance in a given course.

5. REFERENCES

[1] Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004). School engagement: Potential of the concept, state of the evidence. *Review of educational research*, *74*(1), 59-109.

[2] Kizilcec, R. F., Piech, C., & Schneider, E. (2013, April). Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the third international* conference *on learning analytics and knowledge* (pp. 170-179). ACM.

[3] Sinha, T., Jermann, P., Li, N., & Dillenbourg, P. (2014). Your click decides your fate: Inferring information processing and attrition behavior from mooc video clickstream interactions. *arXiv preprint arXiv:1407.7131*.

[4] Taylor, C., Veeramachaneni, K., & O'Reilly, U. M. (2014). Likely to stop? predicting stopout in massive open online courses. *arXiv preprint arXiv:1408.3*

How quickly can wheel spinning be detected?

Noboru Matsuda
Texas A&M University
4232 TAMU
College Station, TX 77843
Noboru.Matsuda@tamu.edu

Sanjay Chandrasekaran
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
sanjayc@andrew.cmu.edu

John Stamper
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jstamper@cs.cmu.edu

ABSTRACT
We have developed a wheel spinning detector for cognitive tutors that uses a simplified method compared to existing wheel spinning detectors. The detector reads a sequence of the correctness of applying particular skill performed by a student using the cognitive tutor. The response sequence is first fed to Bayesian knowledge tracing to compute a sequence of probability of mastery at each time a skill was applied. The detector uses a neural-network model to make a binary classification for a response sequence into wheel-spinning and none-wheel spinning. To test the accuracy of the detector, we validated the detector using learning interaction data taken from a school study where students used a Geometry cognitive tutor. Human coders manually tagged the data to identify wheel spinning. The results show that the neural-network based detector has high recall (0.79) but relatively low precision (0.25) when combined with Bayesian knowledge tracing that detects mastery cases. The result suggests that the neural-network based detector is practical and has a potential for scalable use such as adaptive online course where cognitive tutors are embedded into online courseware.

Keywords
Wheel spinning; detector; neural network; Intelligent tutoring system; student modeling

1. INTRODUCTION
Cognitive tutors provide mastery learning on cognitive skills [3]. Mastery learning is controlled by a student-modeling technique called knowledge tracing [2] that computes the likelihood of mastering individual cognitive skills to be learned. The output from the knowledge tracer is used to compute an optimal sequence of training problems in such a way a student will achieve the mastery for all cognitive skills quickly [4].

One of the challenges under the paradigm of model-tracing based mastery learning happens when the student model does not detect a mastery within a reasonable amount of time. From the students' point of view, this means that they are continuously posed problems one after another for considerably long time. This phenomenon is called *wheel spinning* that has been coined by Beck and Gong [1].

Wheel spinning, by definition, means a situation in which a student does not reach to a pre-defined mastery level according to the mastery estimation computed by the knowledge-tracing algorithm. Although some students may eventually reach mastery only after working on a considerably many number of problems, it is not practical to assume that students would be persistent under such situation. When students do not see any improvement in their performance and the system merely provide more problems, then they would quickly get frustrated and lose their motivation. It is therefore quite important to detect wheel spinning as soon as possible. A reliable student-modeling technique to predict wheel spinning is there required.

The goal of current study is to develop a detector that detects a risk of wheel-spinning at an early phase of learning in the context of cognitive tutoring. The simplicity and scalability of the technology is one of the most important issues. We therefore only use response sequences (i.e., a series of 0's and 1's showing the correctness of application of a particular skill performed by a particular student) as an input to the detector in the current study.

A higher level research question is if we can detect wheel spinning at all: Can we detect wheel-spinning only from the sequence of response accuracy? If so, how accurate the detection is? We hypothesize that if teachers can systematically identify the moment of wheel-spinning only by observing the correctness of student's response, then a neural-network model should be able to learn to detect the moment of wheel-spinning in the same way as teachers do.

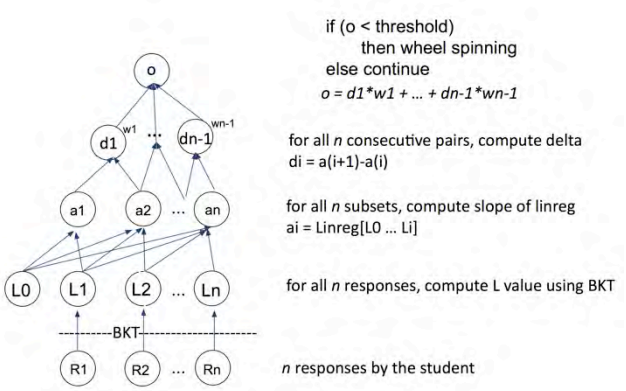
2. THE DETECTOR
Our basis for identifying wheel spinning is to analyze the correctness of student responses for a particular skill. We then attempted to test our hypothesis by comparing the predictions of our detector with examples classified by human coders. We asked two human coders to qualify the student's response data to identify wheel-spinning cases based on our coding manual. Table 1 shows a contingency table showing the agreement between two coders. The inter-coder reliability (the Cohen's kappa) on this final coding is 0.90.

Table 1. Inter-coder agreement of wheel-spinning coding

		Coder 2		
		W	C	Total
Coder 1	W	72	13	85
	C	5	752	757
	Total	77	765	842

Having identified the wheel spinning cases, we attempted to train a neural network to learn a latent pattern in a gradual change in a

sequence of 1s and 0s, representing the first attempt a student has a step for a certain skill.



The input of the NN-based detector is a *response sequence* (denoted as R_1, R_2, …, R_n in the figure) that shows a chronological record of the correctness of skill application made by the student on a particular skill. Each time a new response is observed (i.e., R_n in the figure), the response sequence is fed into the Bayesian Knowledge Tracer (BKT) to update a predicted mastery level up to the point of the latest response observation (denoted as L_1, L_2, …, L_n).

The first part of our neural network computes the change in the predicted mastery level represented as a slope of a linear regression model with the L value as a dependent variable and the opportunity count (i.e., i in L_i) as an independent variable. The slope of this line represents how gradual the student's learning has been. The second part of the neural network computes the deltas for each of the consecutive slope values. Students who are consistently learning have deltas greater than or equal to 0, because overall the trials that those students make forward progress. However, in the case of wheel spinning, the slopes decrease more often than they increase.

The output from the neural network is a weighted sum of the delta values (in the second hidden layer) representing the likelihood of wheel spinning. We train the neural network to learn weights for each delta values in such a way that the output less than zero indicates a potential of wheel spinning and the smaller the output value the more likely the student would wheel spin. The neural network updates weights using back propagation to converge on a set of weights that minimize the classification error during the training.

3. RESULTS

We used the dataset "Cog Model Discovery Experiment Spring 2010" in the study called "Geometry Cognitive Model Discovery Closing-the-Loop", taken from DataShop[1]. This dataset contained 5385 student-skill responses. Among 5385 student-skill response sequences, there are 2883 response sequences that have more than and equal to 5 responses. We filtered out response sequences with less than 5, because there would not be enough attempts to determine wheel spinning. Out of 2883, there are 842 response sequences that do not reach to the mastery according to BKT (hence potentially wheel spinning). In these 842 response sequences, there are 122 unique students and 44 unique skills included.

For our validation study, we decided to use only student-skill response sequences that had greater than or equal to 10 opportunities, because we were trying to find out the best number of opportunities to predict from 5 to 10. After filtering out instances with less than 10 attempts, we were left with 141 student-skill response sequences. We then randomly dropped one response sequence to have 140 student-skill response sequences for a 10-fold cross-validation. On the 9 folds training data, each of the skill-specific neural networks was trained until it classified training instances with the minimum classification errors. The accuracy of the prediction was computed as an overall average across 10 cross-validations. We computed a precision and recall score for each 10-fold-validation, along with a corresponding F1 score. Figure 3 shows precision, recall, and F1 (which is 2*P*R/(P+R) where P and R shows precision and recall respectively) scores for N = 5 to 10.

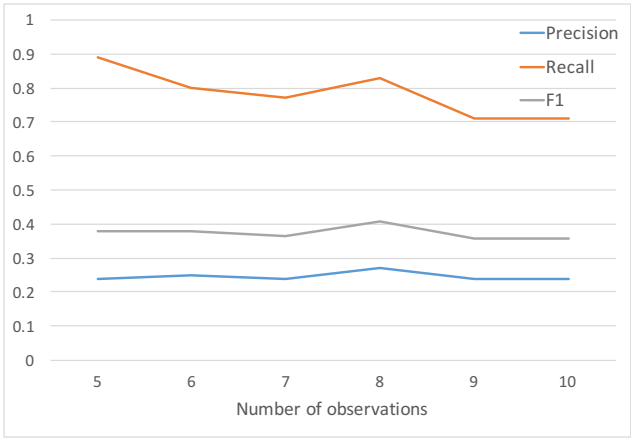


Figure 1. The precision, recall, and F1 scores computed on the first N response observations.

ACKNOWLEDGMENTS

The research reported in this paper has been supported by National Science Foundation Award No. 1418244.

REFERENCES

[1] BECK, J.E. and GONG, Y., 2013. Wheel-Spinning: Students Who Fail to Master a Skill. In *Artificial Intelligence in Education*, H.C. LANE, K. YACEF, J. MOSTOW and P. PAVLIK Eds. Springer Berlin Heidelberg, 431-440. DOI= http://dx.doi.org/10.1007/978-3-642-39112-5_44.

[2] CORBETT, A.T. and ANDERSON, J.R., 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User Adapted Interaction 4*, 4, 253-278.

[3] RITTER, S., ANDERSON, J.R., KOEDINGER, K.R., and CORBETT, A., 2007. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review 14*, 2, 249-255.

[4] VANLEHN, K., 2006. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education 16*.

[1] https://pslcdatashop.web.cmu.edu

Exploring and Following Students' Strategies When Completing Their Weekly Tasks

Jessica McBroom, Bryn Jeffries, Irena Koprinska and Kalina Yacef
School of Information Technologies, University of Sydney, Sydney, NSW 2006, Australia
jmcb6755@uni.sydney.edu.au, {bryn.jeffries irena.koprinska kalina.yacef}@sydney.edu.au

ABSTRACT
In this paper, we explore methods of analysing data obtained from an autograding system involving weekly tasks and a finite set of possible strategies for completing these tasks. We present an approach to handling partially missing information and also investigate the usefulness of a sliding window rule mining technique in following changes in student strategy over time.

Keywords
Mining student behaviour and strategies, autograding system

1. INTRODUCTION
Teaching activities are often not offered in a linear way: it is sometimes useful to provide students with several choices of task, or to provide a gradual approach to learning by allowing a choice of tasks of varying difficulty. Maximum points could be achieved through implementing all the hard tasks, but students unsure of their ability might choose to take a more gradual approach, starting with the easy task and working up. We wish to understand how students manage their learning when presented with such choices by analysing the order in which students attempt such tasks. We investigate the following research questions: What strategies do students take in attempting the different tasks each week? Are there differences between the strategies of the regular and advanced students? In this work we report on several techniques applied to the data collected through an autograding system in a university database course. Our main contribution is in showing how to represent and mine data from student attempts of tasks with different levels of difficulty.

2. DATA
The data comes from weekly programming tasks in a third-year database course with students in a regular stream [2] (n=92), and an advanced stream [3] (n=20). Part of the assessment, for 10% of the final grade, was a set of weekly programming tasks for which students were required to implement various algorithms in Java and submit these implementations using the PASTA online submission platform 0. Tasks included skeleton code and unit tests, and students were encouraged to write and test their implementations locally before submitting. Once submitted to PASTA, the unit tests were applied again, and students received automated feedback of the outcomes of these tests. Students then had the option of submitting a revised attempt, or trying another of the three tasks, until the submission deadline had been reached. Each week there was a choice of three tasks with different levels of difficulty - easy, medium and hard. More marks were allocated for the more difficult tasks: 4 points for hard, 3 for medium, and 2 for easy tasks. Partial implementation of any task received 1 point. The data extracted from PASTA consisted of the marks for every student's attempt on each task.

3. STRATEGIES
There are 16 possible strategies that can be taken by a student for each weekly set of tasks: the 15 possible permutations of Easy (E), Medium (M) and Hard (H) tasks attempted, and no attempt at any task (None). Figure 1 shows the relative frequency of the different strategies taken by all students each week, and in total across all weeks. We labelled each strategy according to the order in which the tasks were completed. So, for instance, in the strategy EH a student completes that week's Easy task first, followed by the Hard task. Note though that this information is imperfect: students were only awarded marks for the most difficult task completed and had access to the unit tests at home, so may have completed multiple tasks while only submitting the most difficult of these. In addition, due to dependencies in tasks in some weeks, certain completion orders were forced. For example, in some weeks the medium task extended the easy task, so students were required to complete easy before medium. However, the most common strategies according to our data are None (30%), E (31%), EM (11%), EMH (15%), EH (4%), H (6%). The remaining 4% is a mixture of the other combinations with support less than 1%, including some where easier questions attempted later: we saw at least one instance of EHM, ME, MH, HE, HM and HME. Two strategies, MHE and HEM, were not observed at all.

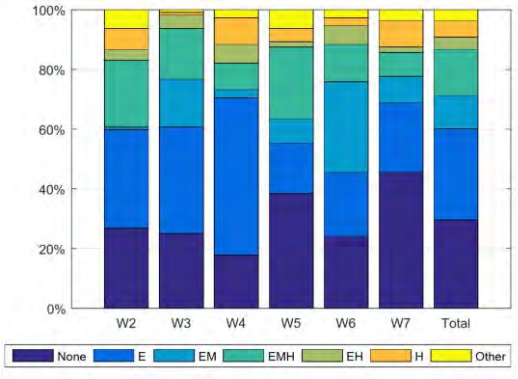


Figure 1. Relative frequency of strategies used by students in each week, and in total across all weeks

4. CLUSTERING
Since students had been allowed to test their code at home, we did not have access to perfect information about the order in which they completed the tasks. We therefore clustered students based

only on the highest difficulty task completed each week, ranking difficulties from 1 (easy) to 3 (hard). E.g., <3, 2, 3, … > would represent a student who completed the hard task in Week 2, the medium task in Week 3 and the hard task in Week 4. Using this representation we applied the *k*-means algorithm with *k*=5 (determined empirically). Cluster centroids are shown in Figure 2.

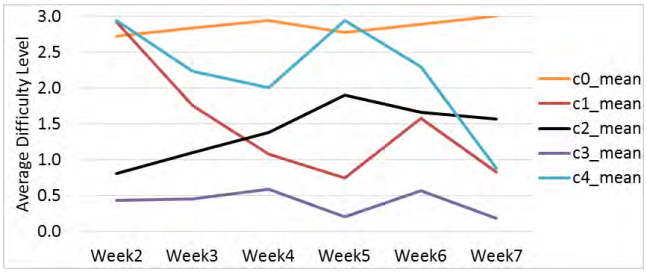


Figure 2. Average highest difficulty of tasks completed by students in each cluster, each week

We complemented the cluster analysis with the information on completion order which involved student submissions with and without potentially missing information. For example, if a student's strategy was EMH, then they definitely completed all three tasks. However, if a student's strategy was H, then they may have only completed the hard task, or they may have completed all three at home and only submitted the hard task. Since strategies with missing information were less frequent, we took the mode weekly strategy for each cluster as shown in 1, which allowed us to still compare student strategies despite the missing information.

Table 1. Mode weekly strategy per cluster. Last column shows proportion of regular and advanced (in parentheses) students.

Cluster	W2	W3	W4	W5	W6	W7	%(adv)
0	EMH	EMH	EMH	EMH	EMH	EMH	9(50)
1	EMH	EM	E	None	EM	None	13(0)
2	E	E	E	E	E	None	18(20)
3	None	None	E	None	None	None	45(15)
4	EMH	EM	E	EMH	EM	E	15(15)

We note that in some weeks there may have been dependencies between tasks that are ignored in this analysis. This limitation notwithstanding, we can broadly summarise behaviour in each clusters. Cluster 0 students complete the hardest task every week, by starting from the easy task and gradually progressing to the hardest task (EMH strategy). Cluster 1 students start well in Week 1 but then gradual drop in the difficulty of the completed tasks towards Week 7. Cluster 2 students start poorly but improve gradually, completing mainly easy tasks. Cluster 3 students consistently make very few submissions, and only of the lowest difficulty. Cluster 4 students generally perform well, often working through tasks of increasing difficulty but not always completing the medium or hard tasks. We speculate that Cluster 3 students may be investing little effort due to the relatively low weighting of the weekly tasks, while Cluster 4 students may have run out of time or found the later tasks too difficult to complete.

5. SLIDING WINDOW RULE MINING

To find trends in changes of strategy we looked for association rules X → Y in which X occurred before Y in time, since only these rules are likely to be of use. We further restricted our analysis to periods of three week. We extracted length-3 itemsets by using a sliding 3-week window over each student's strategy vector. Hence a student's 6-week behaviour vector <2EMH 3EM 4E 5EMH 6EM 7E> would generate 4 item sets <1EMH 2EM 3E>, <1EM 2E 3EMH>, <1E 2EMH 3EM>, <1EMH 2EM 3E>. This process is similar to rule mining in time-series subsequences [1], but here we encode the time into each item to allow us to use traditional association rule techniques.

Table 2. Highest-confidence rules found using length-3 sliding window rule mining technique

Rule	Support	Confidence	Lift
1None,2None → 3None	14%	85%	2.70
1EMH,2EMH → 3EMH	5%	62%	4.63
1EMH,2EM → 3E	3%	57%	2.00
1None,2E → 3E	3%	45%	1.58
1None,2E → 3None	3%	45%	1.43

From these item sets (n = 448) we searched for rules 1*a*,2*b* → 3*c* where *a*, *b* and *c* were the strategies used in consecutive weeks. The 5 highest confidence rules are shown in Table 2. The first rule shows that the likelihood of not attempting a task was very high if the student had not submitted two previous tasks. The second two rules suggest a student is likely to work through all three tasks progressively if they did so in the previous two tasks. Most other rules indicate that many students' strategies were on the borderline between completing the task only or none at all. Our technique was limited by task dependencies; we believe its effectiveness could be improved if applied to data without these deficiencies.

6. CONCLUSION

We have demonstrated how clustering can be applied to data from tasks in which students have choices between several activities, with a particular focus on handling missing information. We have also demonstrated how rule mining can elucidate trends in behaviour over a window of time, though the application of this technique was limited by missing information. These techniques were both limited by variability in dependencies in the different tasks, but still demonstrate how useful knowledge can be extracted from such data.

7. ACKNOWLEDGMENTS

This work was funded by the Human-Centred Technology Cluster of the University of Sydney.

8. REFERENCES

[1] Shokoohi-Yekta, M., Chen, Y., Campana, B., Hu, B., Zakaria, J., Keogh, E., 2015, Discovery of Meaningful Rules in Time Series. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1085-1094

[2] INFO3404: Database Systems 2 (2015 - Semester 2). https://cusp.sydney.edu.au/students/view-unit-page/uos_id/125755/vid/309912. Accessed: 2016-05-20.

[3] INFO3504: Database Systems 2 (Adv) (2015 - Semester 2). https://cusp.sydney.edu.au/students/view-unit-page/uos_id/125790/vid/309914. Accessed: 2016-05-20.

[4] Radu, A. and Stretton, J. PASTA, School of Information Technologies, University of Sydney, http://www.it.usyd.edu.au/~bjef8061/pasta/. Accessed: 2016-05-20

Identifying Student Behaviors Early in the Term for Improving Online Course Performance

Makoto Mori
Department of Computer Sciences
Florida Institute of Technology, USA
mmori2013@my.fit.edu

Philip K. Chan
Department of Computer Sciences
Florida Institute of Technology, USA
pkc@cs.fit.edu

ABSTRACT
To study the correlation between student behavior and performance, we propose using high-level behavior features and a random forest algorithm. Considering a course with 10 periods, our results indicate that our models can reach 70% accuracy in the first period and 90% in the first 5 periods and starting to study earlier is important in individual behaviors and behavior combinations.

1. INTRODUCTION
The main goal of this study is to identify student behaviors in the first half of the semester that are correlated to strong performance so that we can provide feedback and encourage more appropriate behavior. The contributions of our study include: (1) we introduce *high-level* behavioral features derived from the course syllabus and sequential patterns; (2) we propose a random forest algorithm with cross-validation; (3) considering a course with ten periods, our empirical results indicate that our models can reach at least 70% accuracy from behavior features in the first cumulative period and 90% from features in the fifth cumulative period; (4) our approach can identify both important single behavior and behavior combinations. Our empirical results indicate that starting to access course materials early (a *high-level* feature) is important in individual behaviors and behavior combinations.

2. RELATED WORK
Many studies, e.g. [5], generally use how frequent activities occur and how long activities take as main features in their models. We call such features low-level features. Besides low-level features, related studies [4, 6] propose sequence of activities as features that come from a sequential pattern mining algorithm [4]. Further, Jo et al. [2] measure the interval of login sessions to find the regularity of login interval. Coffrin et al. [2] analyze the ordering of materials used in a course. We call features that not only simply measuring frequency and duration of activities as *high-level* features. For learning algorithms, many related studies, e.g. [8], use a single learning algorithm to predict student performance. However, Elbadrawy and Studham [3] propose using linear multi-regression, which is a weighted sum of multiple linear regression models. Many related studies perform performance prediction based on analysis using student activities from the entire term, which does not allow intervention during the term. Some related studies, e.g. [3], use non-behavior features such as quiz or assignment scores in their model. A number of studies only analyze individual behaviors separately. However, some studies analyze behavior combinations. Elbadrawy and Studham [3] use a weighted sum of multiple linear regression models, each of which can be considered as a behavior combination. Kinnebrew and Biswas [6] use SPAM [4] to identify important sequence of learning behaviors. Our approach uses high and low-level behavior features early in the term with an ensemble learning algorithm to identify both important single behaviors and behavior combinations.

3. APPROACH
In this study we focus on three steps. The first step is to generate features that can represent students' behavior. The second step is to use a machine learning algorithm to find correlations between behavioral features and performance. The third step is to identify important behaviors from the learned models.

3.1 Generating Features
Based on our experience, we identify *low-level* features that characterize the amount of different activities. Activities include number of logins, number of videos watched, number of questions asked and so on. ASRs (Active Student Responding Exercises) are questions that are embedded in the instructional video and students enter their answers after watching the video.

For *high-level* features, we focus on measuring beyond just "how frequent" or "how much" from the log files. For example, a motivated student would likely schedule a regular study time. To measure how regular a student studies, we first identify the day of the week that the student studies the most. For example, if a student studies most on Wednesdays, the student is quite regular in using Wednesday for studying. We then divide the frequency of the most studied weekday (e.g. Wednesday) by the frequency of the weekday (e.g. Wednesday) in the behavior period. The course syllabus has due dates and test dates. We generate features of student behavior with respect to those dates. For example, number of days the student studies before a test, number of days to submit a test before it is due. The syllabus also specifies when materials are released. We generate features that measure how soon the student starts accessing the released materials. We use SPAM [4] to identify high-level features based on behavior sequences. SPAM finds sequential patterns that meet the minimum support and maximum gap constraints. Support is the count of a sequence, while gap is the number of "wide cards" between items in a sequence.

3.2 Random Forests with Cross Validation
To improve effectiveness, we propose using the random forest algorithm [16] which builds multiple less-correlated decision trees and combines the classifications from individual trees. The random forest algorithm has two key parameters: forest size (number of trees) and feature subset size (number of features that can be considered in each node). To find a suitable combination of forest size and feature subset size, we vary the two parameters, build a forest, estimate the quality of the forest via cross validation (by splitting the training set), and select the parameter combination that yields the most accurate forest.

3.3 Identifying Important Behaviors
Given a random forest, we identify the most frequent feature used in the root nodes as the most important single behavior. In a random forest, the root of each tree is selected from a random subset of all the features. Hence, the most frequent feature in the root nodes is most likely to be the most important behavior.

Considering a single behavior might not be sufficient, we desire to study behavior combinations that are correlated with higher performance. Consider a forest that has n trees, we calculate a quality score for each feature combination that appears in the top two levels of a tree. The score of feature combination f_i in tree r is the number of positive examples $P_r(f_i)$ divided by the total number of examples $T_r(f_i)$ for this combination. The score of a feature combination $S(f_i)$ in the forest is the sum of scores from the trees: $S(f_i) = \sum_{r=1}^{n} \frac{P_r(f_i)}{T_r(f_i)}$.

4. EXPERIMENTAL EVALUATION

Our main task is to find important behaviors in the first half of the term that correlate with an above average score on the final exam. Also, we identify behaviors that we can encourage later, instead of just asking students to perform better on assignments and tests. Within the first half of the term, we would like to study how early we can identify important behaviors that estimate performance accurately. We divide the first half of the term into multiple periods (e.g. weeks). Features are generated from behavior in period 1 through *k*. We call such periods as "cumulative" periods.

This study analyzes BEHP5000 "Concepts and Principles of Behavior Analysis" that was offered in 2013 at Florida Institute of Technology. We obtained data for 110 students from the course. Our evaluation criterion is prediction accuracy on the test set. Two thirds of students are randomly selected to form the training set and the rest of students are in the test set. To generate sequential patterns with the SPAM algorithm, we use 70% as the minimum support and 2 as the maximum gap.

To compare the effectiveness of our proposed approach with existing approaches, we select a decision tree learning algorithm without and with rule post-pruning [7]. We also choose the original random forest algorithm [1] that uses 100 as the forest size, and log2**M* as the feature subset size, where *M* is the number of features. We use k=5 in the k-fold cross-validation for our random forest algorithm. For each k-fold cross-validation, we vary the forest size from 99 to 999 and the feature subset size from log2**M* to 55.

4.1 Predicting Performance on Final Exam

According to Figure 1, random forest with k-fold cross-validation is the most accurate among the four algorithms. Random forest based models are more accurate than other algorithms. Our approach reaches 74% of accuracy in the first cumulative period, and 90% of accuracy in the fifth cumulative period.

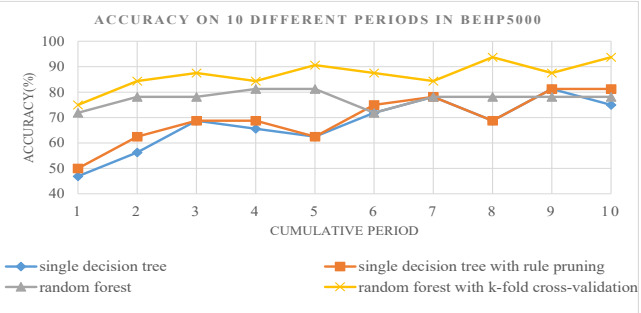


Fig. 1. Accuracy of 4 algorithms from 10 cumulative periods.

4.2 Important Student Behaviors

In the first half of the semester the most frequent feature is *days_after_unit_release* and appears in every cumulative period. This behavior measures, after the unit materials have been released, how many days the student takes to start accessing the materials. The behavior indicates how early a student starts to study, and hence, how motivated the student is. The second most frequent feature is *total(asr_times)* which appears 3 times. This behavior measures the number of times a student attempts ASR, which tries to improve student engagement and understanding of concepts presented in videos. More ASR attempts indicate a student is more engaged and yields deeper understanding.

The most frequent behavior combination is *total(days_after_unit_release)*>x and *test_submit_before_due* <=y which is marked in blue. Both features are high-level features. *total(days_after_unit_release)* represents how early the student starts to access to the unit material after it has been released. *test_submit_before_due* represents how early students submit test before the due date that is stated in the syllabus. Both features are highly related to study motivation of students. Smaller x and larger y values indicate higher motivation. That is, we expect *total(days_after_unit_release)* "<" x and *test_submit_before_due* ">=" y would indicate a highly motivated student. However, we found *total(days_after_unit_release)* ">" x and *test_submit_before_due* "<=" y is the most frequent. In other words, the student begins accessing the materials later and submits the test later, which is counter intuitive. One possible reason is that the behavior combination identifies a small group of students who are smart, therefore, they start studying later and submit test later. Another reason is that the behavior combination appears in cumulative periods 2 and 3, which include less data for the student behavior, therefore, the behavior combination might be less reliable.

Due to space limitation, further details can be found at: `cs.fit.edu/~pkc/papers/edm16long.pdf`.

5. REFERENCES

[1] Breiman, L., 2001. Random forests. *Machine Learning*, 45(1), pp.5-32.

[2] Coffrin, C., Corrin, L., de Barba, P. & Kennedy, G., 2014. Visualizing patterns of student engagement and performance in MOOCs. In *Proc. Intl. Conf. on Learning Analytics & Knowledge* (pp. 83-92).

[3] Elbadrawy, A., Studham, R.S. and Karypis, G., 2015. Collaborative multi-regression models for predicting students' performance in course activities. In *Proc. Intl. Conf. on Learning Analytics & Knowledge* (pp. 103-107).

[4] Ho, J., Lukov, L., & Chawla, S. 2005. Sequential pattern mining with constraints on large protein databases. In *Proc. Intl. Conf. on Management of Data (COMAD)* (pp. 89-100).

[5] Jo, I., Kim, D. & Yoon, M., 2014. Analyzing the log patterns of adult learners in LMS using learning analytics. In *Proc. Intl. Conf. Learning Analytics & Knowledge* (pp. 183-187).

[6] Kinnebrew, J. & Biswas, G., 2012. Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution. Proc. *Intl. Conf. Educational Data Mining* (pp. 57-64).

[7] Mitchell, T., 1997. *Machine learning*. McGraw-Hill.

[8] Seaton, D.T., Bergner, Y., Chuang, I., Mitros, P. and Pritchard, D.E., 2014. Who does what in a massive open online course? *Comm. of the ACM*, 57(4), pp.58-65.

Time Series Analysis of VLE Activity Data

Ewa Młynarska
Insight Centre, University College Dublin, Ireland
ewa.mlynarska@insight-centre.org

Derek Greene
Insight Centre, University College Dublin, Ireland
derek.greene@ucd.ie

Pádraig Cunningham
Insight Centre, University College Dublin, Ireland
padraig.cunningham@ucd.ie

ABSTRACT

Virtual Learning Environments (VLE), such as Moodle, are purpose-built platforms in which teachers and students interact to exchange, review, and submit learning material and information. In this paper, we examine a complex VLE dataset from a large Irish university in an attempt to characterize student behavior with respect to deadlines and grades. We demonstrate that, by clustering activity profiles represented as time series using Dynamic Time Warping, we can uncover meaningful clusters of students exhibiting similar behaviors even in a sparsely-populated system. We use these clusters to identify distinct activity patterns among students, such as Procrastinators, Strugglers, and Experts. These patterns can provide us with an insight into the behavior of students, and ultimately help institutions to exploit deployed learning platforms so as to better structure their courses.

Keywords

Learning analytics, Data mining, Moodle, Time series, VLE

1. INTRODUCTION

The availability of log data from virtual learning environments (VLEs) such as Moodle presents an opportunity to improve learning outcomes and address challenges in the third level sector. We propose representing a student's efforts as a complete time-series of activity counts. We analyse yearly anonymised Moodle activity data from 13 Computer Science courses at University College Dublin (UCD), Ireland, and seek to identify patterns and relationships between more than one attribute that might lead to a student failing a course. A major potential benefit of this would be to introduce mechanisms identifying issues in the learning system early during the semester, supporting interventions and changes in the way in which a course is delivered.

A large amount of previous research in this area relates to different activity types, which are most predictive for a single dataset [1, 3]. This makes it difficult to generalise those methods to systems where the type and volume of Moodle activity can vary significantly. In order to facilitate the performance prediction on less structured systems, we need methods incorporating multiple features to deal with the sparsity problem. As a solution, we present a method for mining student activity on sparse data via Time Series Clustering. We explore the use of Dynamic Time Warping (DTW) as an appropriate distance measure to cluster students based on their activity patterns, so as to achieve clustering indicating more structured activity patterns influencing students' grades. DTW allows two time series that are similar but out of phase to be aligned to one another. To gain a macro-level view regarding whether these patterns occur across all assignments, we subsequently perform a second level aggregate clustering on the clusters coming from each assignment. This results in seven prototypical behaviour patterns (see example in Figure 1), that we believe can lead to better understanding of the behaviour of larger groups of students in VLEs.

2. TIME SERIES ANALYSIS

To perform clustering, the Moodle activity data was transformed into a series of equispaced points in time. In our case, a time series is a three week timeline – from two weeks before a given assignment submission date until one week after the deadline. These timelines were divided into 12 hour buckets of activity counts. We applied k-means clustering using DTW as a distance measure to cluster the timelines for each assignment. For a given number of clusters k, the algorithm was repeated 10 times and the best clustering was selected (based on the fitness score explained below). Due to the fact that DTW is not a true metric, k-means is not guaranteed to converge, so we limited each run to a maximum of 50 iterations. To choose the size of the DTW time window, we ran k-means for *window sizes* $\in [0, 3]$. The results did not conclusively indicate that any single *window size* leads to a significant decrease in cluster grade variance, which is unsurprising. In cases where there are many time series exhibiting little activity, it will be difficult to differentiate between the series and so a larger window size will be more appropriate. Based on this rationale, we believe that *window size* selection should be run for each assignment separately when applying this type of analysis in practice. The fitness function helping in selection of the best clustering needs to take into consideration that two clusters of different sizes might have the same variance value; this issue can be solved by applying a penalty to smaller clusters. We also

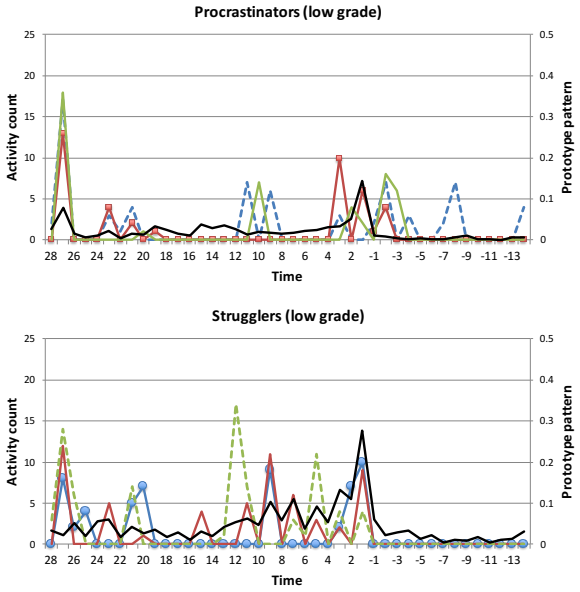


Figure 1: Two of the seven prototype activity patterns that occurred in Assignment #1. The black trend-line represents the prototype pattern. The coloured lines represent the activities of individual students. Negative numbers on the Time axis represent time after the deadline.

would like a "balanced clustering" where the variance of the cluster sizes is as small as possible. Based on these requirements, the fitness score calculation for a clustering generated by k-means consists of three steps:

1. The mean variance of the k-means clustering is calculated using the weighted average of all the clusters' variances, where the weight is based on the size of the cluster. This way the clusterings containing larger clusters with lower variances will be awarded better scores.
2. It is crucial to test the difference between a baseline clustering and actual results to define the significance of the clustering. For that purpose we run multiple random assignments of time series to calculate the expected score which could be achieved by chance for a given number of clusters.
3. To incorporate the baseline comparison in the score, the weighted average variance score from Step 1 is normalised with respect to the random assignment score from Step 2. A good clustering should achieve a low resulting score.

3. DISCUSSION

In our analysis, we took into account 52 two weeks assignments due to their longer and richer time series. We applied the time series clustering methodology described in previous section to the activity data for each of the assignments in the dataset, which are naturally split into two semesters. The Semester 1 clusterings appeared to show a number of frequently-appearing patterns across different courses. To gain a deeper insight into these patterns, we applied a second level of clustering – i.e. a clustering of the original clusters from all assignments. To support the comparison of clusters originating from different modules, the mean time series for each cluster was normalised. Based on the associated assignment scores, these normalised series were then stratified into low, medium, and high grade groups. We subsequently applied time series clustering with $k = 4$ and *window size* 1 to the normalised series in each of the stratified groups. Grade group names chosen by us were motivated by the behavioural pattern of students and some of them were inspired by previous research [2]. This second level of clustering revealed seven distinct prototypical patterns, which are present across multiple assignments and courses: *Procrastinators, Unmotivated, Strugglers, Systematic, Hard-workers, Strategists and Experts.*

The students rewarded with low grades were the second largest group of submissions after medium graded submissions having the smallest average activity per submission. The first out of 3 largest clusters was a group barely active on Moodle, performing submission activity at the deadline only (See Figure 1). As mentioned by Cerezo *et al.* [2], these could be labelled as Procrastinators. The black trend-line on the graph depicts prototype activity pattern and group of time series represents activity of students from the sample cluster. The third biggest group contains those students doing the minimum amount of work and showing larger activity towards the deadline (see Figure 1). The second academic semester courses mostly exhibit similar clusters from the first semester. The percentages indicate that for the Low Grade group, the Strugglers were most common and Procrastinators were less common.

While we did observe significant numbers of outliers, the relevant courses should be considered using a separate analysis to determine whether external factors are at play (e.g. continuous assessment rather than discrete assignments, lack of material provided on Moodle for a specific course). Finally, it is worth exploring anomalous clusters in the context of activity outside that assignment or course. We are currently in the process of extending our research to address the behavioural patterns of knowledge seekers in alternative, more complex learning environments.

Acknowledgments

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

4. REFERENCES

[1] C. Brooks, C. Thompson, and S. Teasley. A time series interaction analysis method for building predictive models of learners using log data. In *Proc. 5th International Conference on Learning Analytics And Knowledge.*, ACM, 2015.

[2] R. Cerezo, M. Sanchez-Santillan, J.C. Nunez, and M.P. Paule. Different patterns of students' interaction with moodle and their relationship with achievement. In *Proc. 8th International Conference on Educational Data Mining*, 2015.

[3] L. V. Morris, C. Finnegan, and S. Wu. Tracking student behavior, persistence, and achievement in online courses. *The Internet and Higher Education*, 8.3:221–231, 2005.

Massively Scalable EDM with Spark

Tristan Nixon
Institute for Intelligent Systems
University of Memphis
365 Innovation Drive
Memphis, TN, USA, 38152
t.nixon@memphis.edu

1. INTRODUCTION

The creation and availability of ever-larger datasets is motivating the development of new distributed technologies to store and process data across clusters of servers. Apache Spark has emerged as the new standard platform for developing highly scalable cluster computing applications. It offers a wide range of connectors to numerous databases and enterprise data management systems, an ever growing library of machine-learning algorithms and the ability to process streaming data in near-realtime. Developers can write their applications in Java, Scala, Python and R. Applications can be run locally (for easy development and testing), and deployed to dedicated clusters or on clusters leased from cloud-computing providers.

2. TUTORIAL

This day-long tutorial will provide a hands-on introduction to developing massively scalable machine learning and data mining applications with Spark. Participants will be expected to follow along with all examples on their own laptops throughout the tutorial, and to collaborate in small groups. All code used in the tutorial will either be taken from publicly available examples, or be available for download from the IEDMS github repository[1], and made available under a very liberal open source license. All examples will be designed to process a modestly sized sample of the KDD cup dataset available from the PSLC DataShop[2].

In advance of the day, participants will be given instructions on how to install and configure Spark and Scala on their laptops, so that they might arrive at the tutorial ready to begin. Throughout the tutorial, participants will be given exercises and problems to solve in small groups. This will give them experience with the material as it is presented and hands-on practice with structuring a distributed application in Spark.

2.1 Outline

The following material will be covered in the course of the tutorial:

- An overview and history of cluster computing and the development of map-reduce
- An example of a very simple map-reduce algorithm (distributed word-count) in Spark
- An introduction to the Spark runtime model, including:
    - Basic import and export operations
    - Resilient distributed datasets (RDDs)
    - RDD transformations and actions
    - How Spark optimizes the execution of distributed computation
- An overview to the different deployment options for Spark, including:
    - Launching and using the interactive spark command-line shell program
    - Running spark programs locally on a single machine
    - Launching a Spark cluster on Amazon Web Services
    - Submitting applications to remote clusters
- An introduction to Spark streaming
- An introduction to SparkSQL and working with DataFrames
    - How to load and manipulate an EDM dataset (KDD cup data)
    - Data representations needed to fit various EDM algorithms
- An introduction to Spark's Machine learning library MLib, including:
    - Transformers and Estimators
    - Chaining transformers into machine-learning pipelines
    - Examples of common EDM algorithms in Spark:
        - IRT algorithms using logistic regression (AFM, PFM, IFM)
        - BKT parameter fitting: (brute-force, HMMs)

Any remaining time will be devoted to discussing potential applications that participants may have in mind for their own data or projects.

[1] https://github.com/IEDMS/spark-tutorial

[2] https://pslcdatashop.web.cmu.edu/KDDCup/

Study on Automatic Scoring of Descriptive Type Tests using Text Similarity Calculations

Izuru Nogaito
KDDI R&D Laboratories Inc.
3-10-10 Iidabashi,
Chiyoda-ku, Tokyo, 102-8460 Japan
+81-3-6678-1609
iz-nogaito@kddilabs.jp

Keiji Yasuda
KDDI R&D Laboratories Inc.
3-10-10 Iidabashi,
Chiyoda-ku, Tokyo, 102-8460 Japan
+81-3-6678-1609
ke-yasuda@kddilabs.jp

Hiroaki Kimura
KDDI R&D Laboratories Inc.
3-10-10 Iidabashi,
Chiyoda-ku, Tokyo, 102-8460 Japan
+81-3-6678-1609
ha-kimura@kddilabs.jp

ABSTRACT

In this paper, we evaluate the automatic scoring of a descriptive type test. In the experiments, three test similarity measures are compared in terms of automatic scoring quality. Two of them are BLEU and RIBES, which are *n*-gram and word-level matching processes respectively, originally used for automatic evaluation of machine translation output. The other similarity process is Doc2Vec, which utilizes distributed representation to calculate the cosine distance. It was finally found that, according to the experimental results, the most efficient process used to calculate the text similarity depends on the type of the question.

Keywords

Doc2Vec, BLEU, RIBES, Text Similarity, auto-scoring

1. INTRODUCTION

Recently, the importance of "21st Century Skills" has been advocated in educational circles. A descriptive type of test is one of the methods to measure this skill; hence, this type of test is becoming more important than a multiple choice test.

In this paper, we carried out experiments on automatic scoring of a descriptive type test. There are two types of methods for automatic descriptive type test scoring. The first method is a similarity-based method, which computes the similarity between a student's answer and a model answer. The second method does not require a model answer; however, it requires several natural language processing (NLP) tools that compute cohesion, coherence, etc. [1]. In this research, we adopt the first approach because our target language for automatic scoring is Japanese and some of the NLP tools are not supported in Japanese. Furthermore, our research partner could provide test items and model answers. In this paper, section 2 describes similarity measures that are used for automatic scoring. Section 3 demonstrates the experiments and their corresponding results, and finally, section 4 describes the conclusions and future work.

2. SIMILARITY MEASURES

In this research, we apply two similarity measures based on surface expression. Both of them were proposed for automatic evaluation of machine translation output. We also apply the similarity measures in a distributed expression to the automatic scoring experiments. In this subsection, we explain these similarity measures.

2.1 Similarity in surface expression

BLEU [2] is proposed for the evaluation of machine translations. It uses *n*-gram matching between a reference sentence and a machine translation output. A sentence that is shorter compared to the reference is penalized in the BLEU score calculation.

RIBES [3] is also an automatic evaluation measure for machine translations. First, it compares the machine translation output with a reference at the word level. Then, it inspects the word order for common words based on the rank correlation coefficient.

2.2 Similarity in distributed expression

Recently, by using deep learning technology, a word or sentence can be converted into a distributed expression that is a vector of several hundred dimensions. According to previous research [4, 5], the cosine similarity between the distributed expressions is fairly close to a semantic similarity. In this research, the gensim [1] version of Doc2Vec is used to build the model that converts the document into a distributed expression.

Table 1: Statistics of the Training Corpus for Doc2Vec

	# of words	Lexicon size
Japanese wiki abstract (WIKI)	29,944,313	1,398,558
Mainichi-News-Paper (1991-2014) (NP)	504,844,192	5,578,327
WIKI + NP	534,788,505	6,376,935

3. EXPERIMENTS

3.1 Experimental settings

Doc2Vec requires a text corpus for model training. For the experiments, we use a Wikipedia corpus (WIKI) and a Mainichi Newspaper corpus (NP). In addition, three models are trained: one using WIKI, one using NP and one using both WIKI and NP. Then, the best model is chosen for each test item in terms of the automatic scoring performance. Table 1 demonstrates the statistics of each particular corpus. In the experiments, we use ten

[1] https://radimrehurek.com/gensim/

test items.

Table 2 Answer Text-Data Specification

Item ID	Topic of question	Question type	Ave. length of student answers (words)	Lexicon size of student answers	Number of students
ID01	Book	Graph reading	112.2	62.5	21
ID02	Fisherman	Summarization	49.7	33.4	21
ID03	Food	Graph reading	96.4	49.0	24
ID04	Fishery	Graph reading	87.8	53.5	22
ID05	Supermarket	Summarization	101.4	59.7	22
ID06	University	Summarization	110.7	71.6	20
ID07	Japanese	Summarization	77.7	46.8	32
ID08	Mail	Summarization	58.9	44.6	42
ID09	Vietnam	Graph reading	57.5	31.2	29
ID10	Beef	Graph reading	90.2	44.2	24
ID01-10	Average		84.3	49.6	25.7

All test items are answered by at least twenty students, aged between 10 and 16 years. Each question has its own target grade. Table 2 demonstrates the data set. In the table, "Graph reading" indicates the situation where the students are asked to describe a fact that can be read from the given graphs. Normally this type of question is a short sentence. Further, "Summarization" indicates the situation where the students are asked to summarize a given text between 300 to 800 words long. In each test item, four model answers are made by four teachers. Each answer is also scored by four teachers. Averaged scores are used as the recorded evaluation results in the experiments.

3.2 Experimental results and Discussion

Figure1 shows the correlation between the subjective score and automatic similarity. For Doc2Vec, we trained models with three conditions: Newspaper corpus only (D2V/NP), Wikipedia corpus only (D2V/WIKI) and both Newspaper and Wikipedia (D2V/NP + WIKI).

The methods that use similarity in surface expression are partly advantageous in the summarization question type. In this type of question, students tend to use the expression in the given question sentence, and the variety of their word choice is small. Thus, the possibility of matching words on the model answer could be high. In fact, the correlation values of BLEU and RIBES for ID02, ID05, ID06, ID07 and ID08 are relatively high.

The methods that use similarity in distributed expression are partly advantageous for the automatic scoring of graph reading questions. In general, the answer for this kind of question has a wide variation of words because students are free to choose their own words.

Both types of results, however, are shown on the graph of reading questions. First, the correlation value from Doc2Vec is better than the other methods for ID03, ID04 and ID10. This is due to the reason described previously. Second, the value of Doc2Vec is inferior for ID01, though it is a graph reading question. In this case, we understand that the corpus used does not share many similar words with the model answer sentences. The result also shows that the Doc2Vec similarity sometimes also works as a complemental similarity.

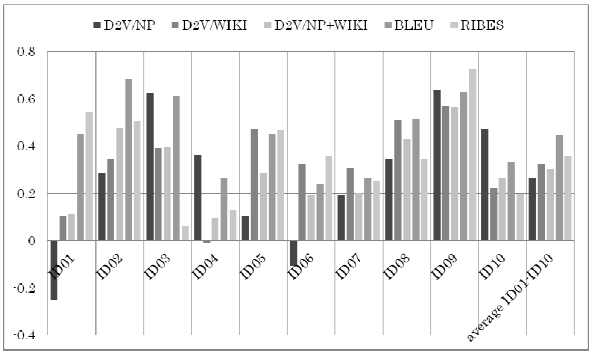


Figure 1 Correlation between subjective score and automatic method

4. CONCLUSIONS AND FUTURE WORK

For automatic scoring, we compared the Doc2Vec, the BLEU, and the RIBES similarities. In the case where the answers include a wide variation of words among students, the method using distributed expression seems to be more advantageous.

In future work, we will conduct research to use several similarities in a complementary way. We will also compare several methods, including the method using cohesion and coherence [1] that is described in the introduction section as a second method.

5. ACKNOWLEDGMENTS

This work uses model answers, student's answers, and scoring data that came from the Lojim clam school. (http://lojim.jp/).

6. REFERENCES

[1] Scott A. Crossley, Danielle S. McNamara.: Cohesion, coherence, and expert evaluations of writing proficiency, Proc. of the 32nd annual conference of the Cognitive Science Society, pp. 984-989, 2010.

[2] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J.: BLEU: a method for automatic evaluation of machine translation, in Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL), pp. 311–318 (2002)

[3] Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, Hajime Tsukada.: Automatic Evaluation of Translation Quality for Distant Language Pairs, Conference on Empirical Methods on Natural Language Processing (EMNLP), Oct. 2010.

[4] Tomas Mikolov and Kai Chen and Greg Corrado and Jeffrey Dean.: Efficient Estimation of Word Representations in Vector Space, http://arxiv.org/pdf/1301.3781.pdf

[5] Quoc Le, Tomas Mikolov.: Distributed Representations of Sentences and Documents, http://arxiv.org/abs/1405.4053

Equity of Learning Opportunities in the Chicago City of Learning Program

David Quigley*, Ogheneovo Dibie, Arafat Sultan, Katie Van Horne, William R. Penuel, Tamara Sumner
University of Colorado Boulder
Boulder, CO 80309-0594
*david.quigley@colorado.edu

Ugochi Acholonu, Nichole Pinkard
Digital Youth Network
2320 N Kenmore Ave
Chicago, IL 60614

ABSTRACT

A novel method for understanding the equity of extracurricular learning opportunities within a regional learning ecosystem is presented. We apply the ecosystems concepts of abundance, richness, and evenness to understand the distribution of learning opportunities within the Chicago City of Learning. This analysis highlights the differences in learning opportunities across different neighborhoods the city. This article includes discussion of the ways these analyses can be used as a starting point for understanding city-wide informal learning communities.

1. INTRODUCTION

This work uses computational approaches to understand the spatial distribution of informal learning opportunities available to youth within the Chicago City of Learning (CCOL), a unique partnership and infrastructure built around supporting youth access to learning opportunities outside of school. Local organizations list their program offerings on the CCOL website and place them in one or more of eleven learning areas such as sports, science, or design. Youth access the site to browse and sign up for these programs. Our aim is to understand the degree to which these afterschool and summer opportunities are accessible to youth. The accessibility of programs relative to where youth live is a matter of *spatial equity* [4].

This research reports on the first year of efforts by CCOL members to document summer informal learning opportunities in Chicago, which resulted in over 4500 searchable learning opportunities. We developed a novel theoretical framework, inspired by concepts from the study of biological ecosystems, that draws on concepts of species richness, abundance, and evenness, and extends these concepts to characterize learning opportunities in a geographic space. We developed data mining approaches for operationalizing these concepts, drawing on data collected through the CCOL system. We present the theory, data mining approaches, and results on a specific question of interest: How are learning activities distributed across different neighborhoods in Chicago?

2. THEORETICAL FRAMEWORK

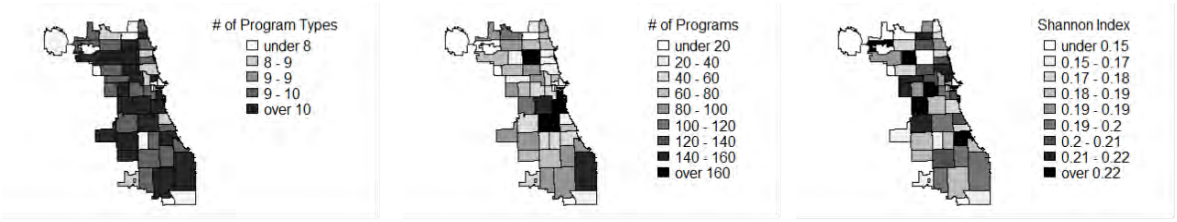
This framework extends Barron and colleagues' descriptions of learning ecologies as linked contexts that provide youth opportunities for learning (e.g. [1]). Human and ecological systems are constantly adapting to changing conditions, including conditions brought about by human activities. Resilient natural ecosystems - that is, ecosystems that have the capacity to adapt to a wide range of unexpected changes - are ones that have both an abundance of organisms and diversity of species [5]. Abundance refers to the number of organisms of a particular species in an ecosystem. Species diversity can be measured in two different ways: species richness and species evenness. Richness is a measure of the number of different kinds of organisms present in a particular area. Evenness measures the relative abundance of each species, or how close in numbers each species in an area are to the others.

These ideas about ecosystems have direct relevance to the study of youths' learning opportunities at the scale of a city. Young peoples' learning pathways are embedded within larger ecosystems of opportunity (e.g. [2]), and these concepts help describe those ecosystems. As in nature where all individual organisms are unique, each program is unique in the learning opportunities it provides to young people. Here, richness, abundance, and evenness refer to program offerings in different neighborhoods, where each individual program is analogous to an individual organism in an ecosystem, a program type is analogous to a species, and a neighborhood is considered a distinct ecosystem.

3. DATA SOURCES AND ANALYSIS

Our team analyzed programs offered through the CCOL website during the summer of 2014, from June 1st to September 30th. We extracted two pieces of information about each program: the program type and the program location. Program type refers to the eleven categories assigned within the CCOL system. Program location is the address of the program as entered by the hosting organization. We normalized the address of each program into a consistent format. We analyzed 3,931 face-to-face scheduled programs at 755 unique locations within the city limits of Chicago.

Program richness provides us with a way to characterize the diversity of opportunities, namely the degree to which program offerings of many different types are accessible from a particular neighborhood. This is determined for each zip code by counting the number of program types that have at least one program hosted in that area. Program abundance refers to the total number of unique programs within a given zip code. Program evenness allows us to measure the degree to which programs of a particular type may predominate



(a) Richness (b) Abundance (c) Evenness

Figure 1: Heatmaps of richness, abundance, and evenness metrics for zip codes in Chicago

in a neighborhood. This measure considers both the types of programs that are accessible and the overall number of programs of each type. We calculated evenness using the Shannon index, the same formula for species evenness in the study of ecosystems [3]. The Shannon index gives an evenness score from zero to one.

4. RESULTS

Figure 1a shows the richness metric - the number of program types with at least one program offering - for each zip code. Our analysis shows that many zipcodes exhibit high richness, with 39 out of the 59 zipcodes having programs spanning 9 or more of the 11 possible program types. Only one zipcode had a single program type being offered.

While many zipcodes exhibit richness, program abundance and program evenness tell a different story. Figure 1b shows the abundance - the total number of program offerings across all program types within each zip code. Here, we see that many of the programs are clustered in certain areas within the city. The large number of programs just south of downtown in particular highlights a hub of programs at cultural institutions such as museums. Other areas, such as the lakefront zip codes north of downtown, host fewer local programs on the CCOL site. Figure 1c shows the program evenness - demonstrated by Shannon index metrics - for each zip code. The indecies in all zip codes are relatively low (0 - .234), showing that all areas' offerings are skewed towards certain categories, rather than hosting a strong representation of programs of all types. In addition, program evenness has a degree of variance between zip codes in the city. Areas west of downtown show slightly better evenness scores than many of those to the south. This metric helps shed further light on the abundance figures shown in 1b. Though the area immediately south of downtown has high measures of abundance, the evenness scores in those same zip codes are lower than scores found in other parts of the city.

5. DISCUSSION

This work establishes a strong understanding of the distribution of learning programs across the city of Chicago. In some areas, cultural institutions are providing many programs in their area, which can skew the evenness metrics in those areas. In others, there are simply relatively few programs being offered. These results illustrate the utility of a data-driven ecological framework for analyzing the distribution of informal learning opportunities within a large urban environment. As the abundance, richness, and evenness heatmaps illustrate, no one metric is sufficient, as each captures different aspects of the larger ecosystem. These three measures, when visualized through the heatmaps in figure 1, provide a concise way to understand distribution of different learning opportunities across the city.

It is important to note the limitations of this approach. First, we used zip codes as our distinct ecosystem boundaries. Some zip codes cover large spaces and have odd shapes, so the presence of a program within that zip code is only a rough proxy of accessibility. Local transit infrastructure can have a significant impact on how well a learner can access a program, even if that program is hosted on the other side of the city. Also, this analysis covers only the first summer of operations of the the CCOL. As such, it is very likely that many learning opportunities taking place in churches, community centers, and other locales are not yet represented in the system. Thus, this analysis presents a single snapshot of only a portion of the total opportunities available to youth in the city.

6. ACKNOWLEDGMENTS

We would like to thank the Chicago Community Trust, the University of Colorado Boulder, and the entire CCOL team for supporting this research. This work would not have been possible without the generous technical support and data sharing provided by the CCOL team.

7. REFERENCES

[1] Barron, B., Gomez, K., Pinkard, N., and Martin, C. K. *The Digital Youth Network: Cultivating digital media citizenship in urban communities*. MIT Press, 2014.

[2] Holland, D., and Lave, J. Social practice theory and the historical production of persons. *Actio: An International Journal of Human Activity Theory*, 2 (2009), 1–15.

[3] Shannon, C. E., and Weaver, W. *The mathematical theory of communication*. University of Illinois press, 1998.

[4] Talen, E. School, community, and spatial equity: An empirical investigation of access to elementary schools in west virginia. *Annals of the Association of American Geographers 91*, 3 (2001), 465–486.

[5] Walker, B., and Salt, D. *Resilience thinking: sustaining ecosystems and people in a changing world*. Island Press, 2012.

Novel features for capturing cooccurrence behavior in dyadic collaborative problem solving tasks

Vikram Ramanarayanan
Educational Testing Service R&D
90 New Montgomery St, #1500
San Francisco, CA
vramanarayanan@ets.org

Saad Khan
Educational Testing Service R&D
600 Rosedale Road
Princeton, NJ
skhan002@ets.org

1. INTRODUCTION

Research shows that complex interactive activities such as team work and collaboration are more effective when participants are not only engaged in the task but also exhibit behaviors that facilitate interaction [5]. Successful collaboration is often manifested in what is known as "entrainment" or convergence between the participants of such collaboration. In the educational context, entrainment between collaborators or between student and the tutoring system is important in understanding learning dynamics, learning gains and student performance in different learning environments [6]. Recently Luna Bazaldua et al. demonstrated a statistically significant synchronicity of cognitive and non-cognitive behavior between dyads engaged in online collaborative activity [1]. However, in their study participants were not able to see each other and only interacted over a text-based chat interface. This is an important point to note since the ability to converse face-to-face can significantly impact the nature of the dyadic interaction. Therefore, in this paper we focus on behavioral patterns of emotional expressions between dyads during face-to-face conversation through a video conferencing system. Our hypothesis is that dyads engaged in face-to-face collaborative activity demonstrate a significantly different pattern of behavior as opposed to nominal dyads who are artificially paired up with each other. Notation-wise, we use the term nominal dyad or artificial dyad interchangeably to mean two subjects whose data are analyzed as if they were interacting dyadically, but were actually not.

Explicitly modeling temporal information in such dyadic interaction data is important because each person's emotional state or behavior need not stay constant over the course of the interaction – they could get fatigued over time, or be more nervous at the very beginning (resulting in repetitive, cyclic fidgeting behavior), but gradually settle into a comfort zone later, as they get more familiar with the task and each other. For similar reasons their body language and emotional state can also fluctuate over the time series. However, current feature extraction approaches that aggregate information across time do not explicitly model temporal cooccurrence patterns; consider for instance that one person's emotional state – joy – generally follows his interlocutor's emotional state – say neutral – in a definitive pattern during certain parts of the interaction. Capturing such patterns might help us (i) explicitly understand the predictive power of different features (such as the occurrence of a given pair of emotions) in temporal context (such as how often did the emotional state of one person in the dyad occur given the previous occurrence of another emotional state of the other person in the dyad), thus allowing us to (ii) obtain features that are more interpretable on visual inspection. We would like to take an initial stab at bridging this gap in this paper. Specifically, we propose to adapt a feature based on histograms of cooccurrences [4] that was developed earlier for analyzing a single time-series (say, from one person), and extend it to the case of dyads (see Figure 1). The feature models how different "template" emotional states of one person in a dyad co-occur within different time lags of a "template" emotional states of the other person in the dyad over time. Such a feature explicitly takes into account the temporal evolution of emotional states in different interaction contexts.

2. DATA

2.1 The Tetralogue CPS Platform

We used an online collaborative research environment developed in-house – the Tetralogue [2, 1]. The participants, who may be in different locations, interact through an online chat box and system help requests (selecting to view educational videos on the subject matter). The main avatar, Dr. Garcia, introduces information on volcanoes, facilitates the simulation, and requires the participants to answer a set of individual and group questions and tasks. A second avatar, Art, takes the role of another student who shows his own answers to the questions posed by Dr. Garcia, in order to contrast his information with that produced by the dyad. Twenty-six subjects participated in this study and were paired in dyads using random selection.

3. ANALYSES AND OBSERVATIONS

In order to observe how well HoC features capture dyadic behavior, we randomly extracted 100 time-intervals (each 10 seconds long) from the post-processed and synchronized feature streams for all 26 subjects. We then computed HoC features for each of these intervals for each subject, respectively. Now recall that in this pool of subjects, each subject has one true dyad with whom they completed the Tetralogue task collaboratively. We hypothesize that the HoC features computed for true dyads will be significantly different as compared to the HoC features computed between artificial or nominal dyads (who did not actually engage in a dyadic interaction). We found that the distances computed between HoC features extracted from true dyads were significantly lower ($p \approx 0$) than those of distances between HoC features computed on artificial dyads. This finding suggests that (i) not only do true dyads engaged in a

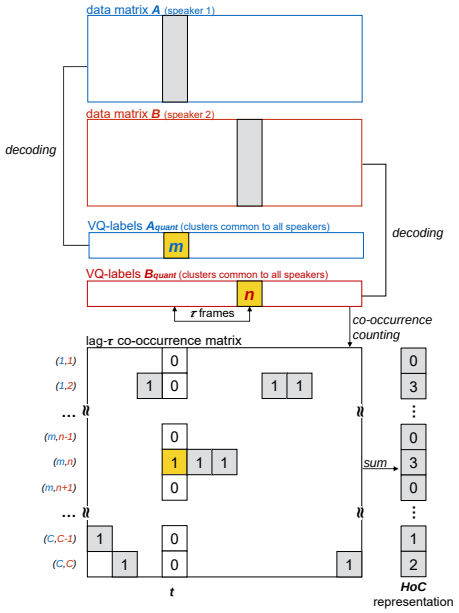


Figure 1: Schematic depiction of the computation of histograms of cooccurrences (HoC) (adapted from [3]).

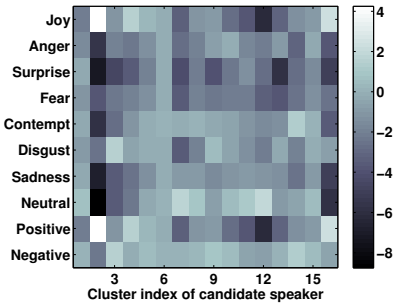


Figure 2: Schematic illustrations of the emotion feature clusters computed for *all* speakers. Each column represents an emotional cluster centroid, which is a particular distribution of emotional state activations. There are 10 dimensions that describe an emotional state, represented by different rows. The colors represent the odds, in logarithmic (base 10) scale, of a target expression being present (typically range: $[-5,+5]$).

collaborative interaction exhibit specific characteristic patterns of emotional state cooccurences that clearly sets them apart from artificial dyads, but (ii) such HoC features allow us to capture these differences in an effective manner.

Figures 2 and 3 gives us some more insight into why these features perform well. Figure 2 depicts the 16 cluster centroids computed on (and therefore common to) all speakers. Notice that each column of Figure 2 represents one cluster centroid, comprising different relative activation of different emotions – for instance, cluster 2 represents an emotional state with a higher activation of joy and positive emotion, while cluster 6 represents a more neutral emotional state, encompassing an equal (and approximately zero) activation of all emotions. Recall that these emotion clusters are common to *all* speakers. Figure 3 shows feature distributions of HoC features computed on one particular speaker and his/her actual dyadic partner, and those computed on that same speaker and an artificial dyadic partner. We observe that the feature distributions of the former are more peaky, with specific certain clusters of emotions

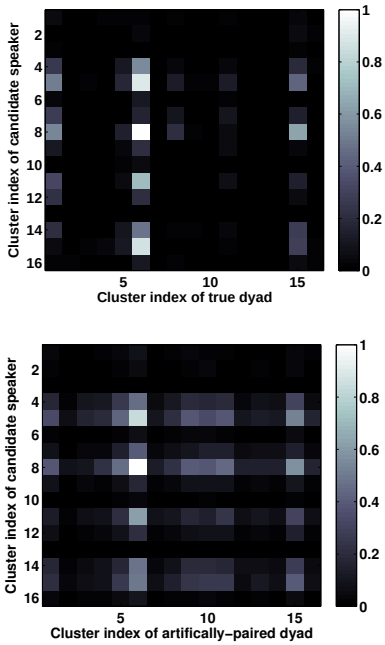


Figure 3: Average HoC feature distributions (across lags) for the true and nominal dyad, respectively, of one particular speaker in the database. The color in the $(m,n)^{th}$ square represents the average normalized activation (between 0 and 1) of cluster m of the speaker represented along the y-axis co-occuring with cluster n of the speaker represented along the x-axis.

co-occurring more often than others. However, in the case of the latter, this distribution is more flat and uniformly distributed. Note that while specific results shown in Figure 3 are particular to the chosen speaker, we observe the aforementioned trends are in general for all speakers. In other words, true dyads display specific patterns of behavioral cooccurrence and synchronicity that are not observed in artificial dyads, and such a HoC feature is helpful in understanding and bringing out these differences.

4. CONCLUSIONS AND OUTLOOK

This paper has made an initial attempt at proposing a novel feature, dubbed histograms of cooccurrences, that captures how often different prototypical behavioral states exhibited by one person co-occur with those exhibited by his/her partner over different temporal lags. We have shown that not only does this feature bring out the differences between dyads and non-dyads, but is also interpretable in that it tells us which behavioral states are most likely to occur in dyads as opposed to non-dyads.

5. REFERENCES

[1] D. L. Bazaldua, S. Khan, A. von Davier, J. Hao, L. Liu, and Z. Wang. On convergence of cognitive and non-cognitive behavior in collaborative activity. In *The 8th International Conference on Educational Data Mining (EDM 2015)*.

[2] L. Liu, J. Hao, A. A. von Davier, P. Kyllonen, and D. Zapata-Rivera. A tough nut to crack: Measuring collaborative problem solving. *Handbook of Research on Technology Tools for Real-World Skill Development*, page 344, 2015.

[3] V. Ramanarayanan, C. W. Leong, L. Chen, G. Feng, and D. Suendermann-Oeft. Evaluating speech, face, emotion and body movement time-series features for automated multimodal presentation scoring. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 23–30. ACM, 2015.

[4] V. Ramanarayanan, M. Van Segbroeck, and S. Narayanan. Directly data-derived articulatory gesture-like representations retain discriminatory information about phone categories. *Computer Speech and Language*, 36:330–346, 2016.

[5] A. A. Tawfik, L. Sanchez, and D. Saparova. The effects of case libraries in supporting collaborative problem-solving in an online learning environment. *Technology, Knowledge and Learning*, 19(3):337–358, 2014.

[6] J. Thomason, H. V. Nguyen, and D. Litman. Prosodic entraiment and Tutoring Dialogue Success. In H. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education, AIED 2013*, pages 750–753. Springer, 2013.

Adding eye-tracking AOI data to models of representation skills does not improve prediction accuracy

Martina A. Rau
Department of Educational Psychology
University of Wisconsin—Madison
1025 W. Johnson St
Madison, WI 53706
+1-608-262-0833
marau@wisc.edu

Zach Pardos
2nd author's affiliation
1st line of address
2nd line of address
Telephone number, incl. country code
2nd E-mail

ABSTRACT
Visual representations are ubiquitous in STEM instruction. Representation skills allow students to use visual representations to learn about concepts. It seems reasonable to hypothesize that we can gather useful information about representation skills from eye-tracking AOI data that assesses how students pay attention to representations. We tested this hypothesis by comparing cognitive models with and without eye-tracking AOI data. Specifically, we used Bayesian Knowledge Tracing and Long Short Term Memory models. We evaluated these models based on their accuracy in predicting students learning of knowledge components that assess representation skills. Eye-tracking AOI data did not improve the prediction accuracy of our cognitive models. We compare our results to prior research to generate hypotheses for future research.

Keywords
Visual representations, intelligent tutoring system, eye-tracking, Bayesian Knowledge Tracing, Long Short Term Memory models.

1. INTRODUCTION
STEM instruction typically uses visual representations that depict to-be-learned content [1]. To learn content knowledge, students have acquire *representation skills*: the ability to use visual representations to learn [2]. Instructional support is most effective if it not only focuses on students' learning of content knowledge, but also on their learning of representation skills [1]. Intelligent tutoring systems (ITSs) have the capability to adapt to the individual student's needs [3]. They do so based on a cognitive model that infers the student's knowledge level based on interactions with the ITS [3]. Hence, the goal of cognitive modeling is to accurately model students' learning in real time [4]. A limitation of this research is that it has mostly focused on students' content knowledge, not on representation skills.

It seems reasonable to assume that we can gather useful information about students' learning of representation skills from their visual attention to representations [5]. However, most prior eye-tracking research involved relatively simple learning materials; typically expository text paired with one additional visual representation. By contrast, ITSs are more complex. Second, prior research has not focused on using eye-tracking AOI data to model students' learning of representation skills. For example, Conati's research group used eye-tracking data in cognitive models found that it can improve predictions of students' learning of content knowledge [6]. This paper tests the hypothesis that eye-tracking AOI data improves cognitive models.

2. DATASET
We used data from a lab experiment that collected students' eye-tracking data while they worked with an ITS for chemistry for 3h [7]. 117 undergraduates participated in the experiment. For our analyses, we used log data from the ITS and eye-tracking data. To analyze the log data, we constructed a knowledge component (KC) model that relates each problem-solving step to the underlying skill. KCs corresponded to representation skills. To analyze the eye-tracking data, we generated visual attention features that assess how students process the visual representations with areas of interest (AOIs) that correspond to the representations. We also created AOIs for the parts of the screen where students solve problems, for the hint window, and for the periodic table that students could show and hide. We included only logged events and first attempts that were tagged with a KC with more than 30 data points. Our final dataset comprised a total of 30,893AOI and log events.

3. ANALYSES
We used two cognitive modeling approaches: Bayesian Knowledge Tracing (BKT) and Long Short Term Memory (LSTM) models. Both analyses used a 5 fold cross validation scheme which was created by assigning students to folds once.

BKT is the standard cognitive modeling procedure in research on ITSs [8]. We used BKT to evaluate a cognitive model representing performance prediction based on a student's history of incorrect and correct responses to questions of the same knowledge component. Following standard practice, we evaluated different guess and slip equivalence classes, which included using a different guess and slip per problem or per step. In previous work [9], separate guess and slip classes at the problem level resulted in a 10% gain in accuracy on ITS dataset. We applied this model to KCs without eye-tracking AOI data and to a version with eye-tracking AOI data. For the latter model, we fit a separate learning rate for each AOI within a problem.

All BKT models were fit with expectation maximization (EM) with max iteration of 100 and epsilon of 1e-6 as stop criteria. The best models in terms of log-likelihood used 40 EM restarts with initial parameter values. Tor prior these were drawn from a uniform random distribution, while the values for learn, guess, and slip were capped at 0.40, 0.40, and 0.30 respectively.

LSTM models are a subset of Recurrent Neural Networks (RNN). Recent progress in image classification with convolutional neural networks utilizes its ability to learn features that have more predictive power than manually crafted features (e.g., edge detection), previously the state of the art for image classification. In a similar vein, we used LSTM so that features of eye-tracking AOI data not yet known to be important could potentially be picked up. Therefore, the LSTM in represents a powerful detector to find out if there is a useful predictive signal in our sequences of eye-tracking AOI data.

We used two LSTM variants on RNNs that add a state to the hidden layer called the cell state which allows the network to

more effectively remember actions that occurred in the past when piecing together patterns in sequential input. We compared versions that utilized eye-tracking AOI data to versions that did not. Both LSTM models utilized the identical amount of information as their BKT with-eye and without-eye data counter parts and both trained a separate model per KC. In the case of LSTM models; eyeHeader, problemID-AOI, and Outcome comprised the feature vector. In both LSTM models, there is an instance of training data for every response given by a student. While non eye-tracking models were trained on sequence lengths that extend as long as the longest response sequence, AOI sequences were limited to the most recent N events, where N was defined as the maximum number of responses of any student in the training data + the median number of AOI events per student. This was done so that the data could fit into memory using 8bit signed integer matrices on a single large memory compute node.

4. RESULTS

After the 5 fold cross validation, RMSE was calculated per student. For a baseline reference, the RMSE of predicting the average percent correct for each KC was 0.39062. Models without eye-tracking data performed better than all of the models with eye-tracking data. Among the BKT models, problem was the better choice for assigning guess and slips over stepname, agreeing with prior work on ITS data [9]. Among LSTM models, extending the number of training epochs from 5 to 10 resulted in the most substantial gain of any model when not using eye-tracking but more epochs lead to overfit with the eye-tracking model. LSTMs, given the same problem-id and response data, were better able to leverage the information towards prediction accuracy than BKT, although both relied on a KC model. Differences between predictions were statistically reliable (ps < 0.05), as determined by a paired t-test of squared residuals between all adjacent models in the list with the exception of the LSTM model with 5 epochs and the BKT model with problem-id as guess/slip, which both used eye-tracking AOI data.

5. DISCUSSION

Our results stand in contrast to our hypothesis: using two cognitive modeling approaches, we did not find evidence that eye-tracking AOI data improves the accuracy of the model's prediction. This finding is noteworthy for the following reasons. First, it is counterintuitive because we tend to assume that visual attention is an important factor in assessing representation skills. Second, our finding stands in contrast to prior research on learning with text paired with one additional visual representation, where students view rather than interact with the material. The difference between prior work and our work is that our study used a complex learning environment, where students manipulated visual representations to solve problems. Third, our results stand in contrast to prior work, which found that eye-tracking AOI data can improve the accuracy of cognitive models of students' learning of content knowledge. The difference between prior work and our work is that our cognitive model assessed students' learning of representation skills, which reflects students' knowledge about the content and about visual representations.

One possible explanation is that prior eye-tracking research on learning with simple materials did not assess whether eye-tracking AOI data adds predictive accuracy to log data—because these materials do not generate log data. Second, representation skills may reflect not how students inspect visual representations, but how they use information from the representations to solve problems, which is sufficiently captured by the log data—particularly if the representations themselves are interactive and hence generate log data that can be used in cognitive models. Third, the fact that we modeled representation skills rather than content knowledge may explain why our results stand in contrast to prior work by Conati's group. We used a KC model that was specifically designed to assess students' representation skills. Even if eye-tracking AOI data assesses representation skills, it may simply not improve the accuracy of our cognitive model because the KC model already captures this information.

A limitation of our research results from the fact that the granularity of our AOIs was fairly coarse. Subtle cognitive signals may exist at fine grained resolutions which may require diving into the raw eye-tracking AOI coordinates. A second limitation was the exploration of hyper parameters. While this is always a caveat of any analysis using machine learning, a particular set of hyper parameters may exist which unlocks the predictive utility of the existing eye-tracking AOI data.

In sum, our findings suggest that eye-tracking AOI data does not necessarily add information relevant to students' representation skills, compared to what can be captured by a well-crafted KC model of representation skills. This rationale amounts to a new hypothesis that should be tested in future research: namely that adding representation skills to cognitive models of content knowledge may improve prediction accuracy in the same way as the addition of eye-tracking AOI data would.

6. ACKNOWLEDGMENTS

This work was supported by a grant from the National Science Foundation (IIS: BIGDATA 1547055).

7. REFERENCES

[1] Gilbert, J.K.: 'Visualization: An emergent field of practice and inquiry in science education': 'Visualization: Theory and practice in science education' (Springer, 2008), pp. 3-24

[2] NRC: 'Learning to Think Spatially' (National Academies Press, 2006)

[3] Koedinger, K.R., Corbett, A.: 'Cognitive Tutors: Technology bringing Learning Sciences to the classroom': 'The Cambridge Handbook of the Learning Sciences ' (Cambridge University Press, 2006), pp. 61-77

[4] Baker, R., Siemens, G.: 'Educational Data Mining and Learning Analytics': 'The Cambridge Handbook of the Learning Sciences' (Cambridge University Press, 2014), pp. 253-272

[5] Mason, L., Pluchino, P., Tornatora, M, Ariasi, N.: 'An eye-tracking study of learning from science text with concrete and abstract illustrations', The Journal of Experimental Education, 2013, 81, (3), pp. 356-384

[6] Bondareva, D., Conati, C., Feyzi-Behnagh, R., Harley, J.M., Azevedo, R., Bouchet, F.: 'Inferring learning from gaze data during interaction with an environment to support self-regulated learning': 'Artificial Intelligence in Education' (Springer, 2013), pp. 229-238

[7] Rau, M.A., Wu, S.: 'ITS support for conceptual and perceptual processes in learning with multiple graphical representations': 'Artificial Intelligence in Education' (Springer International Publishing, 2015), pp. 398–407

[8] Anderson, J.R., Boyle, C.F., Corbett, A.T., Lewis, M.W.: 'Cognitive modeling and intelligent tutoring' (Elsevier Science The MIT Press, 1990)

[9] Pardos, Z.A., Heffernan, N.T.: 'KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model': 'User Modeling, Adaption and Personalization' (Springer, 2011), pp. 243-254

MATHia X: The Next Generation Cognitive Tutor

Steven Ritter
Stephen E. Fancsali
Carnegie Learning, Inc.
437 Grant Street, 20th Floor
Pittsburgh, PA 15219, USA
1.888.851.7094 {x122, x219}
{sritter, sfancsali}
@carnegielearning.com

ABSTRACT

MATHia X is the next generation implementation of Carnegie Learning's Cognitive Tutor (CT), a widely deployed, research-based mathematics curriculum that has provided data for many educational data mining studies. While many researchers are familiar with the basic operation of the system, there are several features that may affect analysis and interpretation of data that are less well known. We describe features of MATHia X and CT, as well as aspects of its practical implementation in real-world classrooms, that may be important for researchers using MATHia X and CT datasets.

Keywords

MATHia X, Cognitive Tutor, intelligent tutoring systems, real-world implementation, mastery learning, wheel-spinning

1. MATHIA X & COGNITIVE TUTOR

MATHia X is the next generation platform for Carnegie Learning's Cognitive Tutor (CT) [5], an intelligent tutoring system (ITS) for mathematics used by hundreds of thousands of learners in middle schools, high schools, and universities across the US (and to a lesser extent internationally, e.g., [4]).

MATHia X provides an HTML5/JavaScript, web-based implementation of the Cognitive Tutor technology and mathematics curricula; for our mid-2016 release we will have content for middle school grades 6-8 and Algebra I, with subsequent content covering Algebra II and Geometry. While MATHia X provides a technology and user interface refresh (including a space-themed interface "skin" in the initial release), fundamentally, most user interface and ITS affordances (including fine-grained data collected about learner interactions in the ITS) are essentially the same as they were in the Java-based Cognitive Tutor and MATHia products that have been in use for well over a decade. As such, we expect to continue in our long-standing tradition of partnering with education, educational data mining, and cognitive science researchers on basic and applied research about how students think and learn, as well as to continue providing data to these communities. The present demo explains a number of features common to both our legacy CT product as well as our next generation MATHia X product, many of which are important to data analyses carried by educational data mining researchers.

Datasets from CT are widely used in a variety of educational data mining (EDM) and education research projects, including in a substantial number of papers in the proceedings of the present conference. Many experimental and observational datasets (comprising hundreds of millions of learner actions in CT) have also been made available via the Pittsburgh Science of Learning Center's DataShop repository [3]. While many aspects of MATHia X and CT, such as their use of mastery learning and Bayesian Knowledge Tracing (BKT) are well known, there are many features and details of implementation and context of use that are less well known but important for appropriate analysis of CT (and eventually MATHia X) data. We describe a number of these characteristics here, in the hope that this information can inform EDM researchers' understanding of CT and MATHia X and contribute to future research that uses such data.

2. FEATURES & IMPLEMENTATION

2.1 Basal and Supplementary Use

Carnegie Learning produces text materials in addition to software, and the "blended" product (text and software) is often used as a "basal" curriculum, meaning that it is the primary source of instructional materials for a class. Our recommendation for blended implementations is that the software be used approximately 40% of the time (two class periods/week), with the text materials used for 60% of classroom time. Depending on school schedules, computer availability, and other factors, the amount of software usage varies considerably between schools.

In addition to "basal" usage, some schools use CT as a supplement to other educational materials. Such usage may follow the 60%-40% model, using a different textbook, but most supplemental usage is irregular. One consequence of such usage is that estimates of student knowledge can be highly inaccurate, since students may learn (or forget) substantial amounts in the long gaps between use of the tutor. Some supplementary use is for a specific purpose (e.g., summer school). In both types of implementations, schools may use the software for all students or for only a subpopulation thereof (e.g., those below grade level).

2.2 (Custom) Curricular Structure

Within K-12, there are a variety of main Carnegie Learning curricula: Algebra 1, Geometry and Algebra 2 (the high school sequence) are provided by our legacy CT product; a three-year middle school sequence and Algebra I are provided by the new MATHia X product in its initial release; and Bridge to Algebra, a one-year review of the middle school sequence is also provided on our legacy platform. Soon all of our curricula will be provided on the web-based technology that drives MATHia X. Overall, these curricula correspond to typical US courses. However, depending on state standards and other needs, schools may construct "custom" curricula that incorporate topics from one or more of these prototypical curricula. Custom curricula are popular, and the majority of CT data is now collected within such custom sequences. CT validates custom sequences for redundancies and violations of prerequisites; schools can ignore warnings about violations, but this is rare.

A curriculum consists of a set of modules, which represents a major topic in the curriculum. A full course may contain 6-8 modules. Modules consist of units, which consist of sections. Each section contains a large set of problems. Mastery learning

operates at the section level; students work within a section until they have mastered all associated knowledge components (KCs) (i.e., skills). The next section (or unit, if the section mastered is the final one in the unit) is automatically presented to the student. The module level is different. Although students will automatically progress to the next module when they complete the final section in the prior module, teachers can also "unlock" modules, allowing students to work on any open modules. Thus, at any given time, a student has a single position within a module (representing the current section) but may have positions within multiple modules. This feature is intended to allow movement among topics that do not have a prerequisite relationship.

2.3 Violations of Mastery Learning

Although we say CT and MATHia X implement mastery learning, in practice, there are several cases where students are not asked to work until they complete with mastery. Within each section of a curriculum, we specify a maximum number of problems that will be presented to students (often 25, but this varies, depending on the complexity of problems; for technical reasons, there are also cases where students might be promoted before reaching this maximum). If students complete this maximum without mastering their skills, they will advance to the next section of the curriculum. We call these advances "promotion," and these are flagged and communicated to teachers in our reporting system. The underlying idea is similar to the concept of "wheel-spinning" [1]. If students are not able to master the material in the tutor in a reasonable period of time, then it is likely that, for whatever reason, the tutor's mode of instruction for this topic is not resonating with the student, and so an alternate instructional approach is preferable. The teacher is responsible for presenting the alternative approach. Promotion is not rare; students are promoted from about 12% of sections. Promotions vary quite a bit by section and by student. Teachers also have the ability to manually move a student to a different position in the curriculum. Such placement changes also violate the mastery assumption. They happen for various reasons, most commonly because the teacher wants the student to "catch up" to the placement of the rest of the class. Such mastery learning violations due to placement changes are associated with greater error rates (and greater variability in error rates) over time than those experienced by students in classes that do not violate mastery learning [6].

2.4 Instructional Resources

Many analyses of CT data have looked at help seeking (e.g., [7]). Such work typically considers student use of problem-specific help, which is the only resource that affects CT's assessment of student knowledge. However, there are other sources of assistance available. Each unit has "lesson" content, which provides declarative instruction, worked examples, manipulatives, and topic-related video. A glossary is always available to students, and references to math terms within lesson text or hints are linked to it. Students also often use calculators and communicate with teachers and other students as they use the software.

Step-by-step examples provide another form of assistance. At least one example problem in each unit illustrates the basic problem-solving approach [2]. Unlike "regular" problems, step-by-step examples expose only one possible path through the problem, and text that would be used as a hint in problem solving is automatically presented to students as they go through the step-by-step example. This experience is intermediate between looking at a worked example and problem solving. Students can refer back to the step-by-step example as they work, and work in the step-by-step example is not used to assess student knowledge.

2.5 Non-persistent Student Model

Math knowledge is cumulative, so one expects that new topics incorporate many KCs mastered in earlier topics. Each section in CT and MATHia X monitors a small set of KCs, among the large set that is actually needed to solve problems in the section. While each section does introduce new knowledge, for various reasons, some sections list KCs that have been addressed in previous sections. These KCs take their preset values, not values based on students' prior work. In other words, CT and MATHia X do not assume that such KCs have been mastered. There is little practical consequence to listing such KCs; if the student learned them, CT will quickly recognize that fact, but researchers should be aware that the CT's assessment of skills is always within a section. Since skill values (i.e., estimates of student knowledge of a skill) do not carry over from section to section, researchers should not automatically assume that KCs with identical names in different sections are, in fact, identical KCs for purposes of data analysis.

3. DEMO + THE FUTURE

In this demo, we will exhibit basic problem solving in MATHia X, introducing the Cognitive Tutor technology to those unfamiliar with it and showing the refreshed technology to those already familiar with our products. Carnegie Learning looks forward to broad adoption of the next generation MATHia X software as a part of its blended mathematics curricula. Combining observational data sets from such adoptions with experimental data sets that will be collected by investigators using MATHia X as a platform for research will provide rich data to be mining and analyzed for many years to come in the educational data mining, learning analytics, cognitive science, and other research communities.

4. REFERENCES

[1] Beck, J.E., Gong, Y. 2013. Wheel-spinning: Students who fail to master a skill. In *Proceedings of AIED 2013* (Memphis, TN, Jul. 2013), 431-440.

[2] Hausmann, R.G.M., Ritter, S., Towle, B., Murray, R.C., Connelly, J. 2010. Incorporating interactive examples into the Cognitive Tutor. In *Proceedings of ITS 2010* (Pittsburgh, PA, Jun. 2010), 446.

[3] Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J. 2011. A data repository for the EDM community: the PSLC DataShop. In *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, & R.S.J.d. Baker, Eds. CRC, Boca Raton, FL.

[4] Ogan, A., Walker, E., Baker, R., Rebolledo, G., Jimenez-Castro, M. 2012. Collaboration in Cognitive Tutor use in Latin America: Field study and design recommendations. In *Proceedings of CHI 2012* (Austin, TX, May 2012), 39-48.

[5] Ritter, S., Anderson, J.R., Koedinger, K.R., Corbett, A.T. 2007. Cognitive Tutor: applied research in mathematics education. *Psychon. B. Rev.* 14 (2007), 249-255.

[6] Ritter, S., Yudelson, M.V., Fancsali, S.E., Berman, S.R. 2016. How Mastery Learning Works at Scale. In *Proceedings of the 3rd Annual ACM Conference on Learning at Scale* (Edinburgh, Scotland).

[7] Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R. 2011. Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learn. Instr.* 21 (Apr. 2011), 267-280.

Towards Integrating Human and Automated Tutoring Systems

Steve Ritter, Stephen E. Fancsali, Susan Berman
Carnegie Learning, Inc.
437 Grant St.
Pittsburgh, PA 15219
{sritter, sfancsali, sberman}@carnegielearning.com

Michael Yudelson
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yudelson@cs.cmu.edu

ABSTRACT

We envision next generation learners having access to both automated and human sources of instruction in a variety of learning contexts. In such contexts, it will be most effective if students can be assisted to appropriately navigate between these sources of instruction. For example, human tutors, when helping a struggling student, might benefit from having access to the learning profile an automated tutor possesses on the student, including what the student already knows, detected misconceptions, inferred affective state and details about the student's work with the automated system before requesting human help. Similarly, an automated tutoring system would benefit from knowledge of interactions during human tutoring session. To facilitate student transitions between these types of systems, we need to understand the factors that best aid students in transitioning between such systems. This poster reports preliminary analyses, suggesting that students who are struggling with the course are more likely to take advantage of the optional human tutoring support and that such use is associated with increased course completion rates, regardless of the student's level of preparation.

Keywords

Human tutoring, intelligent tutoring system, blended approach.

1. INTRODUCTION

Intelligent tutoring systems (ITSs) frequently seek to mimic the best practices of one-on-one human tutors to drive improved student learning outcomes in a manner that is both scalable and cost effective. While extensive research considers a learning context in which a student uses an ITS while having a human instructor available (e.g., in K-12 computer labs), little work considers situations in which students use an automated tutoring system like an ITS alone (e.g., in their homes) while having human tutors available optionally for tutoring sessions via online chat. Data collected under such circumstances has the potential to generate important insight into how instructional "hand-offs" should proceed between such instructional modalities as well as general best practices for human and automated tutoring.

This project builds on more than a decade and a half of research on Carnegie Learning's Cognitive Tutor (CT) ITS [1]. The project leverages a unique dataset comprised of detailed learning records for thousands of students taking an online developmental math course. Students had required CT assignments as well as access to an online chat-based human tutoring service. This dataset allows us to explore the reasons that may lead students to choose to seek help from human tutors while using an intelligent tutoring system. The project also heavily draws on extensive work on tutorial dialogue data [2-3], allowing us to understand the human tutoring interactions that lead to the greatest learning gains within this context. At a technical level, the work further extends prior work exploring tutorial dialogue interactions and their automated classification by incorporating new and previously unavailable machine tutor data.

To the best of our knowledge, the proposed approach we are starting to work towards is the first attempt to address the creation and evaluation of an integrated approach to capitalize on the joint compensatory nature and data exchange between computerized tutors like ITSs and human tutors. We expect tools and results to generalize beyond the specific automated and human tutoring systems examined. For example, we expect knowledge gained from this work to inform us about how to better educate teachers about how to assist students in classrooms using the educational software in physical classrooms and how to build better reporting systems for human tutors helping students in a wide variety of educational applications.

As our first step in understanding how students navigate between CT and human tutoring (HT), we were particularly interested in understanding whether the subset of students who chose to use HT differed substantially in their use of CT and in their outcomes from students who did not use HT. In order to understand whether student preparation for the course affects use of HT, we use student performance in the first week of the course as a proxy for their initial ability in the course.

2. DATA

We collected data from two developmental college mathematics courses (one is a prerequisite for the other) deployed online at a degree-granting institution. Each course took place over five weeks, and the assignment for each week consisted of one large CT module. Each of these modules was broken into sections of content that grouped roughly similar problems. The instructional model within CT employs a mastery learning approach, in which, new problems are given until the CT's estimates of the underlying skills surpasses mastery thresholds. New sections of each math course begin every week; our dataset consists of all CT and HT interactions taking place from June 1 to December 31, 2014. The subject population consists of 16,905 CT users, approximately 3,300 of whom opted to request HT help during the selected period. These students produced over 19,000 human-tutored sessions, with an average length of 22 minutes. Students were predominantly adult learners of college age and older.

3. RESULTS

Table 1 shows primary descriptive statistics for these populations. Statistics for both courses were merged for simplicity since they are quite similar. The data indicate that students who opt to use HT struggled with the courses more than students who did not take advantage of HT. Students using HT have a higher assistance score (number of hints plus number of errors) in CT, as opposed to those who did not use HT. Perhaps as a result of asking for more hints and making more errors, students using HT worked more slowly, completing fewer sections per hour. The measure of sections per hour has been previously found to be predictive of overall course achievement [4].

These results are consistent with the idea that students who are struggling with the course are more likely to take advantage of HT. It seems unlikely that use of HT would have strong effects on course-level measures like amount of assistance or completion of sections per hour, since, on average, students who used HT used it fewer than 6 times in a course covering between 25 and 50 topics.

In contrast to these indicators that students using HT struggle with the course is the data showing that such students are more likely to complete sections in the course. That is, despite the fact that students turning to HT struggle with the course, they complete more sections of the course, indicating that HT may have a broad effect on student persistence.

To further investigate this effect, we use performance in the first module in the course as a proxy for students' initial preparation for the course. To better align Course 1 and Course 2, module 1 performance was converted to a z-score relative to the mean for that course and binned. Bin size was set to 0.5 standard deviations. Figure 1 shows means of course completion probability for each bin for users and non-users of HT with the number of students printed next to each point. At all levels of course preparation, students using HT, although, as we have seen, struggling, are more likely to complete the CT course material.

Table 1. CT and HT statistics: means (standard errors).

	Students using HT	Students not using HT
Parameter	**Course 1**	**Course 1**
CT sections attempted	50.25 (0.25)	50.25 (0.25)
CT problems attempted	493.22 (3.39)	359.38 (2.95)
CT assistance score	3003.16 (47.40)	2621.52 (47.70)
CT assistance score per section	62.78 (1.05)	71.36 (1.24)
CT time per student (hours)	35.41 (0.47)	35.85 (0.50)
CT sections mastered per hour	1.57 (0.03)	0.99 (0.02)
HT time per student (minutes)	110.05 (5.14)	N/A
HT utterances per student	352.82 (17.13)	N/A

4. Conclusion

These preliminary analyses provide a basis for understanding the factors that lead students to use HT and for understanding the broad influence of HT on students. These data are suggestive that students who are struggling with mathematics are more likely to use HT. Interestingly, the data are also suggestive that use of HT may have a broad affective influence on students. Despite the relatively small amount of contact with human tutors during the course, it appears that students who take advantage of such contact appear to be more willing to stick with the course and complete more work, despite their struggles with the mathematics.

5. ACKNOWLEDGMENTS

This work is supported by the contract with Advanced Distributed Learning agency of the Department of Defence (award W911QY-15-C-0070).

6. REFERENCES

[1] Ritter, S., Anderson, J.R., Koedinger, K.R., & Corbett, A. (2007). The Cognitive Tutor: Applied research in mathematics education. Psychonomic Bulletin & Review, 14(2), pp. 249-255.

[2] Morrison, D. M., Nye, B., & Hu, X. (2014). Where in the data stream are we?: Analyzing the flow of text in dialogue-based systems for learning. In R. A. Sottilare, X. Hu, H. Holden, & K. Brawner (Eds.) Design Recommendations for Intelligent Tutoring Systems: Volume 2: Adaptive Instructional Strategies and Tactics (pp. 217–223). U.S. Army Research Laboratory.

[3] Rus, V., D'Mello, S., Hu, X., & Graesser, A.C. (2013). Recent Advances in Conversational Intelligent Tutoring Systems, AI Magazine, 34(3):42-54.

[4] Ritter, S., Joshi, A., Fancsali, S.E., and Nixon, T. (2013). Predicting Standardized Test Scores from Cognitive Tutor Interactions. In Proc. of the 6th International Conf. on Educational Data Mining (Memphis, TN, July 6-9, 2013). 169-176.

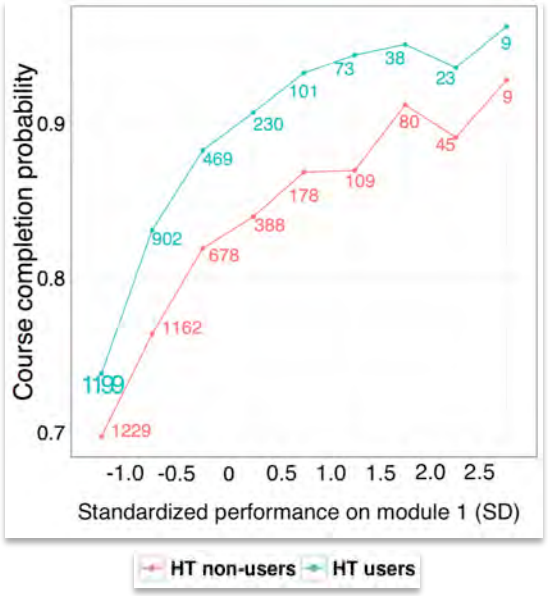


Figure 1. Standardized performance on module 1 vs. overall course completion probability.

Toward Revision-Sensitive Feedback in Automated Writing Evaluation

Rod D. Roscoe
Arizona State University
Rod.Roscoe@asu.edu

Matthew E. Jacovina
Arizona State University
Matthew.Jacovina@asu.edu

Laura K. Allen
Arizona State University
LauraKAllen@asu.edu

Adam C. Johnson
Arizona State University
acjohn17@asu.edu

Danielle S. McNamara
Arizona State University
Danielle.McNamara@asu.edu

ABSTRACT
Revising is an essential writing process yet automated writing evaluation systems tend to give feedback on discrete essay drafts rather than changes across drafts. We explore the feasibility of automated revision detection and its potential to guide feedback. Relationships between revising behaviors and linguistic features of students' essays are discussed.

Keywords

Automated Writing Evaluation; Writing; Revising; Intelligent Tutoring Systems; Natural Language Processing; Feedback

1. INTRODUCTION
Automated writing evaluation (AWE) systems provide computer-based scores and feedback on students' writing, and can promote modest gains in writing quality [1, 2]. One concern is that students receive feedback on their *current* drafts that ignores *patterns of change* from draft to draft. We argue AWE tools should include feedback models that incorporate data on students' revising behaviors and textual changes. These innovations may afford greater personalization of formative feedback that helps students recognize how their editing actions affect writing quality.

This study used Writing Pal (W-Pal), a tutoring and AWE system that supports writing instruction and practice [3, 4]. When submitting essays to W-Pal, students receive scores (6-point scale) and feedback with actionable suggestions for improvement. Scoring and feedback are driven by natural language processing (NLP) algorithms that evaluate lexical, syntactic, semantic, and rhetorical text features [1, 5]. One goal for W-Pal development is feedback that promotes more effective revising [see 4].

2. METHOD

2.1 Context and Corpus
High school students ($n = 85$) used W-Pal to write persuasive essays on the topic of "fame." Most identified as native English speakers (56%) and others as English-language learners (44%).

2.2 Detection and Annotation of Revising
We calculated difference scores between drafts for several NLP measures (via Coh-Metrix [5, 6]). Lexical measures assessed word choice and vocabulary, such as word frequency and hypernymy. Cohesion indices assessed factors such as overall essay cohesion, semantic relatedness (using LSA), and structure.

Human annotation of revisions adapted methods from prior research [7, 8]. Writers can alter their text via adding, deleting, substituting, or reorganizing actions. Human coding of these revision actions showed high reliability ($\kappa = .92$). Revisions can also maintain (superficial edits) or transform (substantive edits) the meaning of surrounding text. Human coding of revision impact on text meaning also demonstrated high reliability ($\kappa = .81$).

3. RESULTS

3.1 Automated Detection of Revising
Essays demonstrated detectable changes in linguistic features from original to revised drafts. Revised essays were longer, included more transitional phrases and first-person pronouns, and were somewhat more cohesive (see Table 1).

Table 1. Linguistic Changes and Correlations with Scores

	Linguistic Change		Correlation with Score Change	
Linguistic Change	***t*(84)**	***p***	***r*(84)**	***p***
Basic				
Word Count	**6.24**	**< .001**	.06	.593
Sentence Count	**4.33**	**< .001**	-.09	.393
Lexical				
Lexical Diversity	-0.28	.781	.17	.124
Word Concreteness	0.83	.410	**.34**	**.002**
Word Familiarity	-0.74	.463	-.01	.954
Word Hypernymy	0.80	.424	**.24**	**.028**
1st Person	**2.09**	**.040**	-.07	.545
2nd Person	-1.06	.294	**-.22**	**.043**
3rd Person	-0.23	.818	-.10	.342
Cohesion				
Connectives	**1.67**	**.099**	.03	.809
LSA Given/New	**2.98**	**.004**	.08	.484
LSA Sentences	0.58	.562	**.24**	**.029**
LSA Paragraphs	**1.86**	**.066**	-.08	.465
Deep Cohesion	0.71	.478	**.18**	**.098**
Referential Cohesion	0.52	.607	.01	.893
Narrativity	1.05	.296	**-.25**	**.023**

Essay quality increased from original ($M = 2.7$, $SD = 1.0$) to revised drafts ($M = 2.9$, $SD = 1.1$), $t(84) = 3.64$, $p < .001$, $d = .19$. Gains correlated with increased concreteness, specificity, objectivity (i.e., fewer 2nd-person pronouns and less story-like), and cohesion. Importantly, the linguistic changes linked to gains were *not* the most typical changes. This finding reinforces the idea that students are not skilled revisers—their revising behaviors can be dissociated from actions that improve the quality of their work.

3.2 Human Annotation of Revising

The most common revisions were additions (47.5%) and substitutions (33.6%). Deletions (15.4%) and reorganizations (2.5%) occurred less often. None of the revising actions were correlated with changes in essay score. This finding reiterates the point that high school students are not necessarily skilled revisers.

3.3 Relationships between Modes of Analysis

The total number of revisions was not related to linguistic changes across drafts (range of *r*s from -.18 to .12). Simply revising *more* had minimal effects. Additions, substitutions, and reorganization had few effects. In contrast, deletions were associated with reductions in narrativity and third-person pronouns. Along with reduced word familiarity, this pattern suggests that students were removing story-like language. Deletions were also associated with reduced given information, semantic similarity across paragraphs, and referential cohesion. Thus, as students removed content from their essays, the cohesive flow of ideas was perhaps hindered. Overall, deletions seemed to be linked to both gains and setbacks in essay quality (see Table 2).

Table 2. Correlations of Revision Types and Linguistic Change

Linguistic Change	Add	Delete	Subst.	Reorg.
Basic				
Word Count	**.29**[b]	**-.36**[a]	-.18	-.10
Sentence Count	**.37**[a]	-.18	-.16	.05
Lexical				
Lexical Diversity	.01	**.26**[c]	-.04	.07
Word Concreteness	.00	**.29**[b]	.08	.06
Word Familiarity	-.04	**-.28**[c]	.15	-.09
Word Hypernmy	-.10	.11	.02	-.18
1st Person	.04	-.11	.11	.07
2nd Person	-.09	-.03	-.05	-.04
3rd Person	-.01	**-.26**[c]	-.07	.00
Cohesion				
Connectives	-.07	.16	.09	-.03
LSA Given/New	-.02	**-.32**[c]	-.07	-.07
LSA Sentences	-.20	-.09	.06	-.12
LSA Paragraphs	.07	**-.24**[c]	-.05	.04
Deep Cohesion	.00	-.11	.07	-.07
Referential Cohesion	-.10	**-.25**[c]	.12	-.03
Narrativity	-.07	**-.34**[a]	-.01	.01

Note. [a]$p \leq .001$. [b]$p \leq .01$. [c]$p \leq .05$.

A final analysis examined revisions by both type *and* impact. As in the previous analysis, the most meaningful linguistic changes were associated with deletions, with substantive deletions appearing to have the strongest influence. Superficial deletions tended to make essays more personalized (i.e., more 1st-person pronouns) and less specific. Substantive deletions tended to make essays shorter, less story-like, more sophisticated in terms of vocabulary, and less cohesive.

4. Discussion

Our results provide evidence that automated tools can detect linguistic changes in students' writing. Formative feedback based on such measures might help students appreciate when and how their drafts evolve over time. For instance, when an increase in narrativity or decrease in cohesion are detected, feedback could flag the edited sections of text so that conscientious students can draw inferences about the impact of their revisions.

Ideally, AWEs should also be able to detect and give feedback on revising behaviors. From the current study, however, it is unclear whether linguistic data could be used to identify such behaviors. With the exception of deletions, students' revising actions did not have a profound impact on linguistic properties.

One solution may reside in keystroke logging [9]. Keyboard and mouse clicks made while interacting with an AWE system may be interpretable with respect to revising. For example, backspace presses may indicate deletion. The use of mouse buttons to select text, along with "CTRL-X" and "CTRL-V" hotkey functions, could signal reorganization. If such tools can be added to AWEs, they may provide real-time measures of writing and revising behaviors that can be explicitly linked to linguistic consequences.

5. ACKNOWLEDGMENTS

This research was supported in part by the Institute for Educational Sciences (IES R305A120707). Opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the IES.

6. REFERENCES

[1] Shermis, M., and Burstein, J. C. (Eds). 2013. *Handbook of automated essay evaluation: current applications and new directions.* Routledge.

[2] Stevenson, M., and Phakiti, A. 2013. The effects of computer-generated feedback on the quality of writing. *Assessing Writing*, *19*, 51-65.

[3] Roscoe, R. D., Allen, L. K., Weston, J. L., Crossley, S. A., and McNamara, D. S. 2014. The Writing Pal intelligent tutoring system: usability testing and development. *Computers and Composition*, *34*, 39-59.

[4] Roscoe, R. D., Snow, E. L., Allen, L. K., and McNamara, D. S. 2015. Automated detection of essay revising patterns: applications for intelligent feedback in a writing tutor. *Technology, Instruction, Cognition, and Learning*, *10*, 59-79.

[5] McNamara, D. S., Crossley, S. A., Roscoe, R. D., Allen, L. K., and Dai, J. 2015. Hierarchical classification approach to automated essay scoring. *Assessing Writing*, *23*, 35-59.

[6] McNamara, D. S., Graesser, A. C., McCarthy, P., and Cai, Z. 2014. *Automated evaluation of text and discourse with Coh-Metrix*. Cambridge University Press.

[7] Faigley, L., and Witte, S. 1981. Analyzing revision. *College Composition and Communication*, *32*, 400-414.

[8] Fitzgerald. J. 1987. Research on revision in writing. *Review of Educational Research*, *57*, 481-506.

[9] Leijten, M., and Van Waes. 2013. Keystroke logging in writing research: using Inputlog to analyze and visualize writing processes. *Written Communication*, *30*, 358-392.

Preliminary Results on Dialogue Act Classification in Chat-based Online Tutorial Dialogues

Vasile Rus, Rajendra Banjade
Department of Computer Science
The University of Memphis
Memphis, TN 38152
{vrus,rbanjade}@memphis.edu

Nabin Maharjan, Donald Morrison
The University of Memphis
Memphis, TN 38152
{nmharjan}@memphis.edu

Steve Ritter, Michael Yudelson
Carnegie Learning, Inc.
437 Grant Street, Suite 918
Pittsburgh, PA 15219, USA
{sritter}@carnegielearning.com

ABSTRACT
We present in this paper preliminary results with dialogue act classification in human-to-human tutorial dialogues. Dialogue acts are ways to characterize the intentions and actions of the speakers in dialogues based on the language-as-action theory. This work serves our larger goal of identifying patterns of tutors' actions, in the form of dialogue act and subact sequences, that relate to various aspects of learning. The preliminary results we obtained for dialogue act classification using a supervised machine learning approach are promising.

Keywords
dialogue acts, intelligent tutoring systems, instructional strategies.

1. INTRODUCTION
A key research question in intelligent tutoring systems and in the broader instructional research community is understanding what expert tutors do. A typical operationalization of this goal of understanding what expert tutors do is to define the behavior of tutors based on their actions.

In our case, because the focus is tutorial dialogues, we model the actions of tutors using dialogue acts inspired from the *language-as-action* theory [1, 7]. According to the language-as-action theory, *when we say something we do something*. Therefore, we map all utterances in a tutorial dialogue onto corresponding dialogue acts using a predefined dialogue act taxonomy, which is described later. It should be noted that automatically discovered dialogue act taxonomies are currently being built [6]. However, we chose to work with an expert-defined taxonomy of dialogue acts, developed by experts based on dialogue and pedagogical theories [5], because it better serves our larger research goals of testing such theories.

2. THE APPROACH
We adopted a supervised machine learning method to automate the process of dialogue act classification. This implies the design of a feature set which can then be used together with various supervised machine learning algorithms such as Naive Bayes, Decision Trees, and Bayes Nets. For automated dialogue act classification, researchers have considered rich feature sets that include the actual words (possibly lemmatized or stemmed) and n-grams (sequences of consecutive words). Besides the computational challenges posed by such feature-rich methods, it is not clear whether there is need for so many features to solve the problem of dialogue act classification.

Our approach is based on the observation that humans infer speakers' intention after hearing only a few of the leading words of an utterance [4]. One argument in favor of this assumption is the evidence that hearers start responding immediately (within milliseconds) or sometimes before speakers finish their utterances ([5] - pp.814).

Intuitively, the first few words of a dialog utterance are very informative of that utterance's dialogue act. We could even show that some categories follow certain patterns. For instance, Questions usually begin with a *Wh*-word while dialogue acts such as Greetings use a relatively small bag of frozen words and expressions.

In the case of other dialogue act categories, distinguishing the dialogue act after just the first few words is not trivial, but possible. It should be noted that in typed dialogue, which is a variation of spoken dialogue, some information is not directly available. For instance, humans use spoken indicators such as the intonation to identify the dialogue act of a spoken utterance. We must also recognize that the indicators allowing humans to classify dialogue acts also include the expectations created by previous dialogue acts, which are discourse patterns learned naturally. For instance, after a first Greeting another Greeting that replies to the first one is more likely. We used intonational clues in our work to the extent that such information is indirectly available to us, in the form of punctuation marks, in typed/chat-based dialogues. We did incorporate contextual clues in our preliminary experiments, e.g. we used as a feature the dialogue act of the previous utterance, but the results did not improve significantly. It is important to note that the present study assumes there is one direct speech act per utterance.

3. THE TAXONOMY
The current coding taxonomy builds on an earlier taxonomy that sought to identify patterns of language use in a large corpus of online tutoring sessions conducted by human tutors in the domains of Algebra and Physics [5]. The taxonomy is considerably more granular than previous schemes such as the one used by Boyer and colleagues [2].

The most recent version of the taxonomy employs two levels of description. At the top level, it identifies 16 standard dialogue categories including Questions, Answers, Assertions, Clarifications, Confirmations, Corrections, Directives, Explanations, Promises, Suggestions, and so forth. It also includes two categories, Prompts and Hints, that have particular pedagogical purposes. Within each of these major dialogue act categories we identify between 4 and 22 subcategories or subacts.

4. EXPERIMENTS AND RESULTS

We have used in our experiments 288 tutorial sessions (containing about 17,537 utterances) between professional human tutors and actual college-level, adult students. These sessions are a subset of of a larger sample of 500 sessions randomly selected from a corpus of 17,711 sessions we obtained from an organization that offers online human tutoring services. Students taking two college-level developmental mathematics courses (pre-Algebra and Algebra) were offered these online human tutoring services at no cost. The same students had access to computer-based tutoring sessions through Adaptive Math Practice, a variant of Carnegie Learning' Cognitive Tutor. It should be noted that students may or may not initiate a tutorial dialogue with a human tutor while attending those courses. This is important to note as there could be a self-selection bias in those tutorial dialogues that we used.

Expert Annotation Process

The 288 sessions we used here were manually labelled by a team of 6 trained annotators, all of whom were experienced classroom math teachers. Each session was first manually tagged by two independent annotators, i.e. they did not see each other's tags. Then, the tags of the two independent annotators were double-checked by a verifier, who also happens to be the designer of the taxonomy. The verifier had full access to the tags assigned by the independent taggers. The role of the verifier was to resolve discrepancies. The inter-annotator agreement for the two independent annotators was Cohen's kappa=0.72 for dialogue acts and kappa=0.60 for dialogue acts and subacts combined.

The agreement was best for Expressives (0.88), Assertions (0.81), Requests (0.78) and worst for Hints (0.2), Clarifications (0.33), and Explanations (0.42).

Results

For space reasons, we summarize the results of our supervised machine learning approach in terms of accuracy and Cohen's kappa relative to the final tag adjudicated by the verifier using a 10-fold cross-validation approach. We only provide results on dialogue act classification (no subacts) for the same space reasons.

The model

Our model for predicting dialogue acts consists of the following five features/predictors: the leading three tokens in an utterance, the last token such as a question mark ('?') at the end of a question, and the length of the utterance. We experimented with other features such as the speaker (student vs. tutor), the position of the utterance in the dialogue, e.g. an utterance at the beginning of a session is more likely a Greeting, the previous dialogue act, but we have not noticed any significant impact on performance relative to the five-feature model mentioned above. More powerful models that do account explicitly for sequential observations are needed, e.g. Conditional Random Fields.

We experimented with our 5-feature model in combination with a number of machine learning algorithms including Naïve Bayes, Decision Trees, and Bayes Nets. We also experimented with sequential models based on Conditional Random Fields but the results, again, were not better. The best results, obtained with BayesNets, are summarized below.

D-Act classification Results

Using all features leads to 67.27% accuracy and Cohen's kappa of 0.58. The speaker does not seem to have an impact as the results accuracy is 66.74%. The same for position, if removed the resulting accuracy is 66.77%. The remaining features are indeed important as if another is removed the accuracy drops significantly below 60.00%.

Our plan next is to annotate more sessions up to 500 and retrain our models. Once the accuracy is at acceptable level, we will use the classifiers to automatically tag tens of thousands of sessions with dialogue acts and subacts. Once the sequences of actions and subactions are available, we will identify patterns of tutor and student actions that related to learning and affect and which could then be used in the development of automated intelligent tutoring systems or in a hybrid system where both human and intelligent tutors co-exist.

Acknowledgments. This research was sponsored by a subcontract to The University of Memphis from Carnegie Learning, Inc., under award W911QY-15-C-0070 by Department of Defense. Any opinions, findings, and conclusions or recommendations expressed in this material are solely the authors'.

5. REFERENCES

[1] Austin, J. L. (1962). *How to do things with words*: Oxford University Press, 1962.

[2] Boyer, K.E., Phillips, R., Ingram, A., Ha, E.Y., Wallis, M.D., Vouk, M.A., & Lester, J.C. (2011). Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden Markov modeling approach, The *International Journal of Artificial Intelligence in Education (IJAIED)*, Vol. 21 No. 1, 2011, 65-81.

[3] Jurafsky, Dan.; and Martin, J.H. (2009). Speech and Language Processing. Prentice Hall, 2009.

[4] Moldovan, C., Rus, V., & Graesser, A.C. (2011). *Automated Speech Act Classification for Online Chat*, The 22nd Midwest Artificial Intelligence and Cognitive Science Conference, Cincinatti, OH, April 2011 (Best Student Paper Award - Honorary Mention).

[5] Morrison, D. M., Nye, B., Samei, B., Datla, V. V., Kelly, C., & Rus, V. (2014). *Building an Intelligent PAL from the Tutor.com Session Database-Phase 1: Data Mining*. The 7th International Conference on Educational Data Mining, 335-336.

[6] Rus, V., Graesser, A., Moldovan, C., & Niraula, N. (2012). *Automatic Discovery of Speech Act Categories in Educational Games*, 5th International Conference on Educational Data Mining (EDM12), June 19-21, Chania, Greece.

[7] Searle, J. R. (1969). *Dialogue Acts: An essay in the philosophy of language*. Cambridge university press, 1969.

SAS Tools for Educational Data Mining

Jennifer Sabourin
SAS Institute
100 SAS Campus Dr.
Cary, NC 27513
1. 919.531.3313
Jennifer.Sabourin@sas.com

Scott McQuiggan
SAS Institute
100 SAS Campus Dr.
Cary, NC 27513
1. 919.531.1119
Scott.McQuiggan@sas.com

Andre de Waal
SAS Institute
100 SAS Campus Dr.
Cary, NC 27513
1. 919.531.6575
Andre.Dewaal@sas.com

ABSTRACT
Researchers in the EDM community have always relied on sophisticated tools to analyze data and build models. As the amount of data that can be collected and stored grows, the need for tools capable of handling "big data" becomes ever more prevalent. SAS® Analytics U is a new initiative for making SAS data analysis and mining tools available for free to educational researchers and instructors. These tools are designed for handling very large data sets and can be run in the cloud, saving researchers valuable time and resources. Furthermore, SAS Analytics U provides a community of SAS educators and learners to share resources and information about SAS tools and techniques.

This tutorial aims to introduce researchers to the tools available through SAS Analytics U and how they can be applied to the field of Educational Data Mining. We will provide an overview of the SAS architecture and provide instruction on the key features of each tool in the suite. We will guide participants through examples using relevant educational data sources to help researchers understand how the tools can be applied to their own work.

REQUIREMENTS: In order to participate in the hands on exercises, please bring a laptop on which you have installed SAS University Edition. The free download is available at http://www.sas.com/en_us/software/university-edition/download-software.html. The download and installation may take up to 1 hour so there will not be time to get set up during the tutorial.

1. TUTORIAL DESCRIPTION
This tutorial will focus on introducing SAS to participants and guiding them through the use of the suite of tools using relevant educational data sets. The tools that will be covered include:

SAS® Programming Language. SAS programming language is a powerful language designed specifically for intensive data analysis. This highly flexible and extensible fourth generation programming language has a clear syntax and hundreds of language elements and functions. It supports programming everything from data extraction, formatting and cleansing to data analysis, building sophisticated models, and generating reports. The SAS programming language is at the heart of the SAS University Edition tools.

SAS® Studio. SAS Studio is the development environment for SAS University Edition and runs through the web browser as well as in the cloud. It offers a powerful GUI interface that allows novice programmers to interact with data and perform analyses without writing any SAS code themselves. However, the SAS code is all generated behind the scenes and is visible to help users learn.

SAS® Enterprise Miner. SAS Enterprise Miner helps users streamline the data mining process to create highly accurate predictive and descriptive models based on analysis of vast amounts of data. It includes innovative algorithms in the areas of statistics and machine learning to enhance the stability and accuracy of predictions, which can be verified easily by visual model assessment and validation. Users build process flow diagrams that serve as self-documenting procedures. These diagrams can be updated easily or applied to new problems without starting over from scratch. In addition to process flow diagrams, Enterprise Miner provides a programming interface for advanced users. Enterprise Miner allows integration with open source software for data manipulation and model comparison, the open standard PMML, and databases for scoring models without data movement.

Additional SAS tools that may be covered if it is of interest to the participants include tools for time series analysis, forecasting, matrix manipulations, and advanced statistics.

2. JUSTIFICATION
Educational data miners rely on computational tools to understand and explore their data. These tools must be robust and flexible in order to allow for innovation. They must be able to handle ever increasing amounts of data. Ideally, they are easy to use by both programmers and non-programmers alike due to the interdisciplinary nature of this research area. Finally, most researchers rely upon tools that are freely available and do not require excessive resources.

SAS University Edition is a new option that addresses many of these needs. This suite of powerful SAS software was made available to all learners for free in May of 2014. SAS Enterprise Miner, Text Miner, and Forecast Server have been available through SAS OnDemand for Academics since late 2010. However, the biggest barrier to adopting new tools is learning how to use them. SAS Analytics U is a community centered around these free offerings and is designed to support SAS learners and educators. This tutorial seeks to introduce participants to these resources and suite of tools and demonstrate how they can be applied to EDM research. The goal is that participants will be able to add another set of tools to their every growing toolbox for conducting EDM research.

3. PRESENTERS
The presenters for this tutorial include both researchers who are active in the EDM community and trained SAS educators who are experienced in leading tutorials of SAS products.

Jennifer Sabourin. Sabourin has a dual role as a research scientist and software developer on the Curriculum Pathways team at SAS Institute. As a research scientist she works on identifying research questions and using machine learning and analytical techniques to improve the efficacy of Curriculum Pathways products. She also serves as a consultant aiding external researchers with using SAS

software to better understand and make decisions from their educational data. As a software developer she works on creating innovative applications for K-12 that are offered at no-cost.

Sabourin received her Ph.D. from North Carolina State University in 2013. Her graduate work focused on data mining and artificial intelligence in game-based learning environments. She has been an active member of the EDM community since beginning her graduate work.

Scott McQuiggan. McQuiggan leads SAS Curriculum Pathways, an interdisciplinary team focused on the development of no-cost educational software in the core disciplines at SAS Institute Inc. Curriculum Pathways includes more than 1,500 resources, tools, and apps for K-12 education used in all 50 states and more than 90 countries around the world. He regularly uses data mining and analytics to better understand the behaviors exhibited in Curriculum Pathways resources and improve the efficacy of the products themselves.

McQuiggan received his PhD in computer science from North Carolina State University, where his research focused on affective reasoning in intelligent game-based learning environments. He also holds an MS in computer science from North Carolina State University and a Bachelor of Science in computer science from Susquehanna University. Scott is co-author of the book, Mobile Learning: A Handbook for Developers, Educators, and Learners.

André de Waal. De Waal is an Analytical Consultant with SAS Institute and his work focuses on teaching users how they can use SAS to best meet their analytic needs. He received his Ph.D. in theoretical computer science from the University of Bristol during 1994. He spent the next year in Germany and Belgium continuing his research in Logic Programming and Automated Theorem Proving. During 1996 he returned to South Africa to take up his position as lecturer at the School of Computer Science and Information Systems at the then Potchefstroom University for Christian Higher Education (which later became the North-West University), where he was later promoted to Associated Professor. During 1999 he became one of the founder members of the Centre for Business Mathematics and Informatics at the same university. He became responsible for the Data Mining Program in the Centre and shifted his research focus to include Neural Networks and Predictive Modeling. He joined SAS Institute in Cary, NC during December 2010 to take up the position of Analytical Consultant in the Global Academic Program.

4. PROPOSED FORMAT

This tutorial will be presented as interactive instructions where users will be guided through the tools using relevant education data with a focus on techniques that are commonly required in the EDM community. The tutorial will also include an overview of SAS and its commitment to education research by a leading SAS executive. We also seek to gain feedback from participants prior to the event so that we can tailor the sessions to specific needs or questions. A tentative schedule (subject to conference timings) is below:

Session 1: Introduction and SAS Studio

9:00-9:15	Introduction – Introduction of presenters and participants and overview of SAS Analytics U
9:15-10:30	SAS Studio
Coffee Break	

Session 2: SAS Studio

11:00-12:30	SAS Studio
Lunch Break	

Session 3: Keynote and SAS Enterprise Miner

14:00-14:30	Keynote – A SAS executive (TBD based on final scheduling) will present an overview of SAS and its commitment to education by discussing tools made available to researchers and products made available to K-12 educators and students.
14:30-16:00	SAS Enterprise Miner
Coffee Break	

Session 4: Participant Requested Instruction

16:30-17:30	Additional Instruction – based on the goals of the participants we will delve deeper into aspects of the tools already presented or introduce additional tools as listed in the tutorial description.
17:30-18:00	Conclusion

In addition to the tutorial, instructional materials will be made available to participants. We will also provide guidance on avenues for further learning through online instruction.

Applicability of Educational Data Mining in Afghanistan: Opportunities and Challenges

Abdul Rahman Sherzad
PhD Student at Technische Universität
Berlin, Germany
absherzad@gmail.com

ABSTRACT

The author's own experience as a student and later as an active lecturer in Afghanistan has shown that the methods used in the Afghan educational systems do not provide students with the minimum guidance needed to select the proper course of study before they enter the national university entrance exam (Kankor). The result is often high attrition rates and poor performance in higher education.

Based on the studies done in other countries, and by the author of this paper through online questionnaires distributed to university students and graduates in Herat, Afghanistan – it was found that proper procedures and specialized studies in high schools can help students in selecting their field of study more systematically. Additionally, there are large amounts of data available for mining purposes but the methods that the Ministry of Education and Ministry of Higher Education use to store and produce their data only enable them to achieve simple facts and figures. Furthermore, from the results it can be concluded that there are potential opportunities for educational data mining application in the domain of Afghanistan's education systems. For instance, predict proper field of study for high school graduates, or, identify first year university students who are at high risk of attrition.

Keywords

Educational data mining; major prediction; student placement; Kankor; Afghanistan education systems; value of information.

1. INTRODUCTION

General education in Afghanistan comprises K-12 (primary, secondary and high school), Islamic studies, Teacher Training, Technical and Vocational schools and institutes which are administered by the Ministry of Education (MoE). The Ministry of Higher Education (MoHE) supervises universities which provide Bachelor's, Master's, and PhD degree programs.

Since the establishment of the new democracy in Afghanistan in 2001, education systems have been going through a nationwide rebuilding process. Despite obstacles, numerous public and private educational institutions were established across the country [2]. The result is a substantial increase in the student enrollment rate, as reflected (see Figure 1).

Every year more than 200,000 students graduate from high schools and around 300,000 participate in Kankor across the country [3].

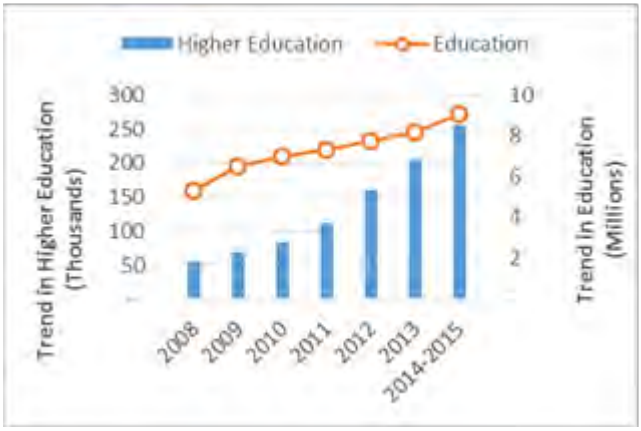


Figure 1. Education and Higher Education enrollment trends.

The MoE and MoHE as the main bodies of education systems in Afghanistan have been trying to standardize the quality of education in order to be able to meet the minimum international standards. In this extremely challenging process, one of the efforts of the MoE and MoHE has been to automate their information through Education Management Information System (EMIS) and Higher Education Management Information System (HEMIS) [6].

The EMIS and HEMIS are able to generate (only) basic statistics (e.g., total number of students and teachers based on gender, geographic location, schools and universities) which are not very helpful in decision making to improve the education systems effectively. For example, '10 million students in schools' is just a number and piece of data without a specific context and further useful information to describe the setting. Hence, these simple facts and figures do not help policy makers to improve the educational settings. For example, one cannot predict proper majors/fields of study (Major) for high school graduates, or, identify first year university students who are at high risk of attrition. This paper will be a new initiative in its kind. The objective is to study the opportunities and challenges of EDM applicability in the Afghanistan education context in order to help educational institutions to better prepare students for their studies in schools and universities.

2. MAJOR RECOMMENDATION

Presently in Afghanistan, school students are not divided into Majors. The author conducted one online survey to public and private university students and graduates, and another survey to computer science students and graduates. A total of 333 people participated in these surveys; 315 agreed that it is more useful if the students are offered specialized studies after grade 9 at school.

Additionally, due to general studies and insufficient orientation on Kankor at schools, the majority of students do not know what Major to choose in the Kankor. This was confirmed by the same online surveys. Besides, in the existing situation, it is found that there are no structural and specialized institutions to provide and guide students on career choices based on their skills and interests. This situation creates a vicious cycle for misappropriating human-capital as the most vital resource for development.

The outcome of these studies [4, 7] can be customized and used to recommend proper Majors to high school graduates prior attending the Kankor, and also while specialized studies are introduced at schools. The following approaches can be used. 1-Assess student performance for 10th, 11th and 12th grades to identify the strengths and weaknesses of the applicants in all the relevant Majors. 2-Since the results of high school grades could be misleading, this research proposes the design of a new standardized test in order to evaluate the interest and capabilities of the applicants through varied 'Yes' and 'No' intelligent questions. 3-Since there are no pre-collegiate courses prior to entering University, it is deemed efficient to evaluate the skills of applicants in the 12th grade through a number of Kankor practice tests. 4-Other simulator (self-assessment) tools as an all-encompassing medium to self-evaluate, capitalize on improving and minimize the identified gaps of candidates and to evaluate the interest and capabilities of the applicants. 5-Of course, social, economic, and literacy status of student's family and other pedagogical factors could be significant for better evaluation and assessment. 6-Divide more than 100 Majors into main major areas including Natural and Social Sciences, Health Sciences, Humanities and Literature, Islamic Education, Fine Arts and Technical Education. 7-Last but not least, consideration of previous Kankor results data during data mining process would lead to better accuracy rate.

3. SUPPORT AT RISK STUDENTS

Most of the students are at risk of dropping out or performing poorly during their higher education studies. One of the main reasons is that the participants randomly select Majors in the Kankor without much knowledge of the requirements and challenges ahead of them and the inventory of their existing knowledge in the relevant field of study. Also, lack of specialized studies at schools is another major reason for attrition and poor performance in higher education. According to the above mentioned online survey conducted by the author among Computer Science students in Herat province out of 227 respondents around 90% did not have the skills and knowledge of basic programming, database, and operating systems, as echoed in (see Figure 2). The result of the survey is showing that one of the major reasons for weak academic performance in higher education is lack of specialized studies in school.

An early counseling intervention solution would be a great support to identify the key factors to improve their academic performance and to decrease rates of attrition through academic counseling, tutorial classes and other supportive programs [1, 5]. This could be achieved with evaluation and comparison of fresh student's data with historical data of senior students. For example, school performance and grades for main prerequisite subjects relevant to their selected Major (i.e. the required score value for Journalism in mathematics might be 2 out of 5, while in Engineering it might be, 5 out of 5), if they attended supportive courses and classes besides school studies, family responsibilities, and other social and extracurricular activities.

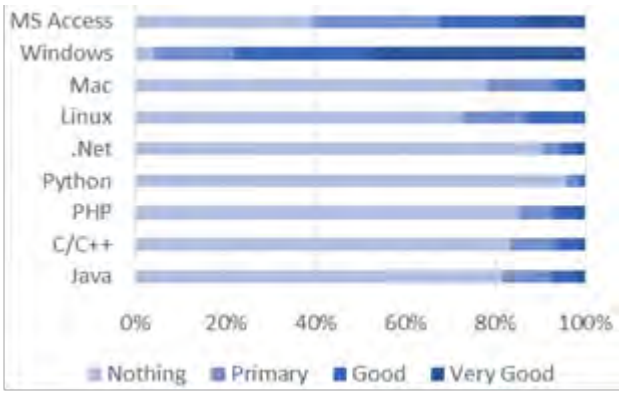


Figure 2. IT skill of computer science students prior Kankor.

4. CONCLUSION

Enrolment trends in Education and Higher Education generates vast amounts of data. With learning and tutoring management systems, the amount of data will be significantly increased either implicitly or explicitly. The main challenge preventing the applicability of EDM is lack of proper data storage and accessibility to data in electronic format. EMIS at MoE and HEMIS at MoHE together could be appointed to provide the raw data for EDM applications to help discern patterns of abilities and behaviors which could be used to help educational institutions.

5. ACKNOWLEDGMENTS

I thank my professors at Technical University of Berlin for their direct and indirect support, and the respondents.

6. REFERENCES

[1] Agnihotri Lalitha, Ott Alexander. 2012. Building a Student At-Risk Model: An End-to-End Perspective. In Proceedings of the 7th International Conference on Educational Data Mining, 209-212

[2] Andishman Mohammad Ikram. 2010. Modern Education in Afghanistan. Maiwand publication

[3] Central Statistics Organization. 2014-2015. Afghanistan Statistical Yearbook: Education Part One. Retrieved June 15, 2015 from http://cso.gov.af/en/page/1500/4722/2014-2015

[4] Emilio J. Castellano, Manuel J. Barranco, Luis Martínez. 2011. Academic Orientation Supported by Hybrid Intelligent Decision Support System, Efficient Decision Support Systems - Practice and Challenges from Current to Future.

[5] Pan Wei, Guo Shuqin, Alikonis Caroline, Bai Haiyan. 2008. Do Intervention Programs Assist Students to Succeed in College?: A Multilevel Longitudinal Study. College Student Journal 42, 1: 90-98

[6] Peroz Nazir, Tippmann Daniel. 2012. Information Technology for Higher Education in Afghanistan: ZiiK Report Nr. 32.

[7] Pratiwi Oktariani Nurul. 2013. Predicting student placement class using data mining. In Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering, 618–621.

Browsing-Pattern Mining from e-Book Logs with Non-negative Matrix Factorization

Atsushi Shimada
Kyushu University
Fukuoka, Japan
atsushi@artsci.kyushu-u.ac.jp

Fumiya Okubo
Kyushu University
Fukuoka, Japan
fokubo@artsci.kyushu-u.ac.jp

Hiroaki Ogata
Kyushu University
Fukuoka, Japan
ogata@artsci.kyushu-u.ac.jp

ABSTRACT

In this paper, we report our work-in-progress study about browsing-pattern mining from e-Book logs based on non-negative matrix factorization (NMF). We applied NMF to an observation matrix with 21-page browsing logs of 110 students, and discovered five kinds of browsing patterns.

Keywords

e-Book logs, pattern mining, non-negative matrix factorization

1. INTRODUCTION

An e-Book system can collect various kinds of operation logs when a page is opened, when the next page is browsed and so on. The analysis of e-Book logs enables teachers to understand how a student browses a given material. However, just giving or showing the logs is insufficient to understand behaviors of students because of their diversity and high dimensionality. In this paper, we apply non-negative matrix factorization (NMF) technique [2], which is known as akin to principal component analysis and factor analysis. In [1], NMF is utilized to extract a Q-matrix[1] from observed test outcome data for n question items and m respondents. In our study, we discover students' browsing patterns, i.e., how they browsed the given material, from e-book logs data for n page browse and m students. Besides, we analyze the relationship between the patterns and quiz scores.

2. E-BOOK LOGS

The e-Book logs were collected from 110 first-year students in an information science course taken in the first semester of the 2015 school year at Kyushu University in Fukuoka, Japan, via BookLooper (Kyocera Maruzen Systems Integration Co., Ltd.). Figure 1 shows samples of e-Book logs. There are many types of operations in logs, for example, OPEN means that the student opened the e-book file and

[1]A mapping of item to skills is termed a Q-matrix

User	Material	Operation	PageNo	Date	Time
X	00000000NLAT	OPEN	0	2014/10/15	9:01:09
X	00000000NLAT	CLOSE	1	2014/10/15	9:01:13
Y	00000000P82P	PREV	25	2014/10/29	10:05:35
Y	00000000P855	NEXT	2	2014/11/19	8:52:47
Z	00000000P84Z	NEXT	9	2014/11/12	9:31:30
...	...	...	...	...	...

Figure 1: Samples of e-Book logs

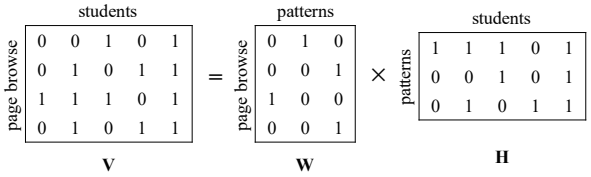


Figure 2: Pattern mining from browsing matrix

NEXT means that the student clicked the next button to move to the subsequent page. The duration of browsing each page can be calculated by subtracting the timestamps between subsequent pages.

3. METHODS

We utilize non-negative matrix factorization (NMF) technique to discover some browsing patterns. NMF approximately decomposes a matrix of $n \times m$ positive numbers V as the product of two matrices:

$$V \approx WH. \tag{1}$$

NMF imposes the constraint that the two matrices, W and H, be non-negative. In our approach, the matrix V, named browsing matrix, is represented by the fact whether a student browsed a page or not. More specifically, we set an element $v_{i,j}$ of the matrix V by

$$v_{i,j} = \begin{cases} 1 & (\text{if } t_{i,j} > th) \\ 0 & (\text{otherwise}), \end{cases} \tag{2}$$

where $t_{i,j}$ is the duration of page i browsed by student j. The decomposed matrices represent two latent relationships: "page browse vs. patterns" given by matrix W and "patterns vs, students" given by matrix H. In the sample of Figure 2,

Table 1: Description of discovered browsing patterns

pattern 1	browse the latter part of pages
pattern 2	browse the former part of pages
pattern 3	browse the middle part of pages
pattern 4	browse the beginning and end part of pages
pattern 5	browse pages between #12 and #15

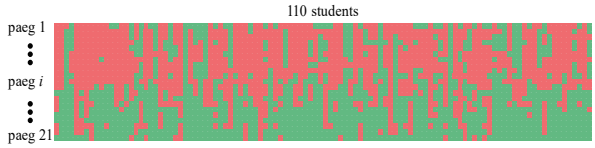


Figure 3: Browsing matrix. The red and green colors denote the value of $v_{i,j}$, red for one, green for zero, where the th was set to be 10 seconds.

browsing patterns are represented by three patterns in W. Meanwhile, H means whether a student has one or more browsing-patterns for a given material. In the experiments, we set the number of patterns to be five.

4. EXPERIMENTS

The browsing matrix used in our experiments were obtained from 110 first-year students. The students were asked to preview the material in advance before the lecture. They browsed the given material of information science which consists of 21 pages with a spread display setting. Therefore, the V is represented by 21-row $\times$ 110-column matrix as shown in Figure 3. The column of V corresponds to a student's previewing history whether he/she spent time at page i longer than th or not, which is calculated by formula (2). In the experiment, we set the th to be 10 (second).

NMF was performed to find five patterns. The decomposed matrices W and H are shown in Figure 4 and Figure 5 respectively. Note that the W is transposed in the figure due to the limitation of page space. Each pattern can be roughly described as Table 1.

The upper part of Figure 5 shows the correspondence between a student and his/her browsing pattern. The red color means that the student has the pattern (for example, the student in the most left column has pattern 2 and pattern 4). After the NMF, we acquired five groups based on consensus clustering technique (refer to literature [3] for more details). The bottom part of Figure 5 is the reordered matrix of W to show the group characteristics efficiently. The group 1 has the pattern 2 more strongly than the other patterns, which means that they spent longer time on the former part of pages.

We compared the student groups with their quiz scores (see Table 2). The quiz (max score = 10) was conducted at the beginning of the lecture. The average score of group 4 was lower than the other groups because the group had no pattern, i.e., they did not browse the material well. On the other hand, the group 2 got the highest score. They had the pattern 5, which corresponds to browsing the pages between #12 and #15. The contents of these pages were related

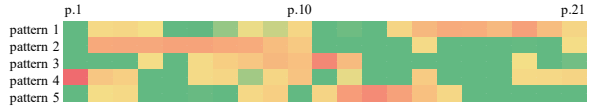


Figure 4: Visualized matrix W. Red parts represent the correspondence to each pattern. For example, pattern 2 denotes that pages from 2 to 10 are well browsed.

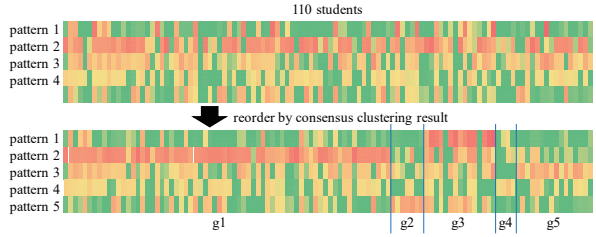


Figure 5: (Top:) Skill matrix visualized by color scale. The red color represents larger value. (Bottom:) Reordered pattern matrix based on consensus clustering result. There are five groups (g1, ..., g5) found by clustering.

Table 2: Average scores of quiz in each student group

student group	g1	g2	g3	g4	g5
average score	6.25	6.95	6.57	5.49	6.00

to the practice exercise to enrich the understanding. We guess that the students in group 2 could work the exercise because they had already understood the basic contents in the material. Therefore they got better quiz scores than the other student groups.

5. CONCLUSION

In this paper, we gave our work-in-progress report about e-Book browsing pattern mining and its potentials to fathom the relationships between patterns and understanding level of contents. In the experiments, we showed a primal result of pattern mining based on NMF. We found out that NMF could provide reasonable decomposed matrices to explain the browsing patterns. In the future work, we investigate the appropriate number of patterns because we predefined the number of patterns in this paper. Besides, we have to consider more effective method to generate a browsing matrix from e-Book logs.

6. REFERENCES

[1] M. Desmarais. Conditions for effectively deriving a q-matrix from data with non-negative matrix factorization. In Proceedings of the 4th International Conference on Educational Data Mining, pages 41–50, 2011.

[2] D. Lee and H. Seung. Learning of the parts of objects by non-negative matrix factorization. Nature, 401:788–791, 1999.

[3] S. Monti, P. Tamayo, J. Mesirov, and T. Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. Mach. Learn., 52(1-2):91–118, July 2003.

How employment constrains participation in MOOCs?

Mina Shirvani Boroujeni, Łukasz Kidziński, Pierre Dillenbourg
Computer Human Interaction in Learning and Instruction
École polytechnique fédérale de Lausanne
{mina.shirvaniboroujeni, lukasz.kidzinski, pierre.dillenbourg}@epfl.ch

ABSTRACT

Massive Open Online Courses (MOOCs) changed the way continuous education is perceived. Employees willing to progress their careers can take high quality courses. Students can develop skills outside curriculum. Studies show that most of the MOOC users are pursuing or have received a university degree. Therefore it is beneficial to consider motives and constraints of this class of participants while designing a course. In this study we focus on time constraints experienced by full-time and part-time employees and students. Surprisingly, activities of students and employees are very similar regarding timing. We found that part-time employees spend more time on forum and are more active during the day. Employees are more active in the evening hours from Monday till Thursday. Based on our findings we suggest course design insights for practitioners.

1. INTRODUCTION

Time management in Massive Open Online Courses (MOOCs) is indispensable for success [2]. Recent studies show that difficulty with keeping up to deadlines is the main obstacle for engaging in a course [1]. Motivated by previous research, we assume that problems with time management are due to either professional constraints or issues with self-regulation [1] as illustrated in Figure 1. In this study we plan to provide a basis for understanding motives and limitations of MOOC participant depending on their employment status. Our general objective is to investigate: **How occupation (student, employee or part-time activity) influences participants time management in MOOC? How is it reflected by their engagement?**

2. DATASET

Our analysis is based on three successive offerings of an undergraduate engineering MOOC offered in Coursera entitled "*Functional Programming Principles in Scala*". The initial dataset contains 133,129 users. However information about the employment status is provided only by 8.7% of the participants. Based on this information, we extracted three categories of users: *full-time employed* (702 users), *full-time student* (110 users), *part-time activity* (66 users). 84% of full-employed participants hold a master or bachelor (45% and 39% respectively) and this ratio for the part-time group is 64% (32% and 32% respectively). Interestingly there is a noticeable percentage (22%) of participants with part-time activity who do not possess an academic degree.

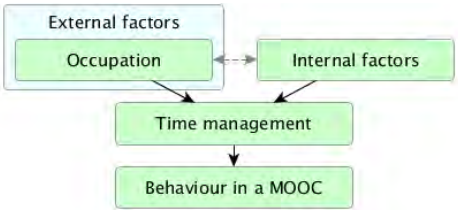


Figure 1: Time management is crucial for success in MOOC. We investigate the influence of occupation on time management.

For the analysis of users' performance we consider two types of events: watching videos and forum activities including viewing the forum (passive events) and writing or voting messages (active events). We extracted a set of features for each user, including final grade, count of forum events (total, active and passive), count of forum messages, average length of messages, count of submitted assignments and average number of attempts per assignment. In addition, we also extracted number of videos watched on different times of the day (Midnight, Morning, Midday, Afternoon, Evening, Night), different days of the week (Monday to Sunday) and different times of each week day. The final set includes 63 feature which were used in the analysis and building a predictive model in the following section.

3. FINDINGS

Q1. Are employed participants more likely to engage in the course? Based on χ^2 test, there is a significant relation between employment status and dropping out ($\chi^2 = 29.06, df = 2, p < 0.01$). According to the test residuals, among the three categories, employed participants are more likely to engage in the course, whereas students are most likely to drop out.

Q2. Do employed participants have higher achievement level? ANOVA on linear model of final grades re-

Proceedings of the 9th International Conference on Educational Data Mining

veals marginal significant difference between grades for students and employed participants (F[1, 810]=3.8, p=0.05): employed participants on average achieved a higher grade compared to the students (70 vs. 63 out of 100).

Q3. Are employed participants more engaged in forum? Total forum activity (active and passive events) by students and employed participants is similar, whereas part-time participants are significantly more active in forum compared to the other two groups (87 vs. 51, Mann-Whitney-Wilcoxon test, W=20516, p<0.01). Similarly number of posts by students and employed participants are not significantly different, while part-time participants have significantly more posts (M=4.6 vs. 1.7 posts, Mann-Whitney-Wilcoxon test, W=21282, p<0.01). Posts by part-time participants are the longest (M=83 words, t=-2.21, df=441.78, p= 0.02) and post by students are the shortest (M=53 words, t=3.14, df=239.35, p<0.01).

Q4. Do employed participant have different weekly pattern of activity? Distribution of videos watched on each week day shows that part-time participants watch more videos during the weekdays, whereas employed users and students are more active during weekends. Sundays and Mondays are the most active days for all groups and the activity level decreases from Monday to Saturday, mainly for employees and student. This trend could be related to the fact that video lectures were released on Sundays.

Q5. Do employed participants have different time distribution of activities? Number of videos watched in different parts of the day shows to be related to the employment status of participants ($\chi^2 = 109, df = 10, p < 2.2e-16$). As shown to Figure 2, employed participants are the most active group during evening hours (F[1, 876]=4.92, p=0.02), students are the most active group during night hours and part-time participants are the most active group during mid-day. Furthermore unlike part-time participants, the activity level of the other two groups is higher during the afternoon and evening compared to the mid-day hours.

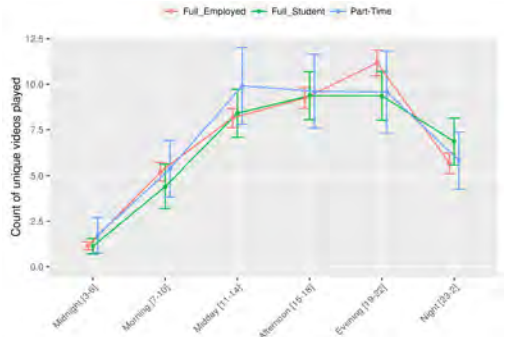


Figure 2: Distribution of number of videos watched at different times of the day.

Further investigations of participants' activity patterns in different days of the week reveals that the observed evening activity peak for the employed participants is related to the working days (Monday to Thursday). On Friday their overall activity level is low and on weekends their activity peak time is shifted to the afternoon hours. Remarkably, all groups are active in the mornings and during the midday. In particular, this could suggest that full-time employees engage in MOOCs during the morning commutes and also during the work day. Nevertheless this finding should be further confirmed in interviews with MOOC participants.

Q6. To what extent can we predict user's employment status based on derived features? In order to predict employment status of participants based on the features described in Section 2, we trained several classifiers including Neural Network, Penalized Multinomial Regression, Random Fores and Support Vector Machine with linear kernel. Using 10-fold cross validation, the highest Cohen's κ (0.45) was achieved by Random Forest classifier.

4. CONCLUSION

Our analysis revealed that employment is reflected by different activity patterns. This confirms our hypothesis that time constraints influence user's participation in MOOCs. Our findings partially confirm previous theories. In particular, higher drop-out rate from MOOCs among students versus employees can be attributed to lower academic and social commitment [3]. This phenomenon can also be linked to better time management of employees (participation in MOOC during the evening just after work) [2]. Further controlled studies should be conducted to discover true causality.

Based on the insight from our analysis, we suggest following design considerations while designing MOOCs courses: **(1) Choose the lecture release day depending on the target audience.** We found that activity of employed participants drops during the weekdays. On the other hand, video release on Sunday make participants work on Monday despite the general lower activity during workdays. Therefore, releasing lectures on Saturday might increase overall activity. **(2) Choose activities convenient for commute time and short sessions.** Our analysis showed activities during potential commuting hours, therefore designing short and mobile-friendly videos and activities could facilitate users engagement during this time. **(3) Choose accurate timing for communication with users,** such as the time when they are most likely to visit the MOOC **(4) Include temporal activity indicators in predictive models,** as time-related features showed to be correlated not only with employment status but also with the success in a MOOC.

5. REFERENCES

[1] René F Kizilcec and Sherif Halawa. Attrition and achievement gaps in online learning. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 57–66. ACM, 2015.

[2] Ilona Nawrot and Antoine Doucet. Building engagement for mooc students: introducing support for time management on online learning platforms. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 1077–1082. International World Wide Web Conferences Steering Committee, 2014.

[3] Vincent Tinto. Dropout from higher education: A theoretical synthesis of recent research. *Review of educational research*, 45(1):89–125, 1975.

Quantifying How Students Use an Online Learning System: A Focus on Transitions and Performance

Erica L. Snow
Center for Technology in Learning
SRI International
333 Ravenswood Ave
Menlo Park ,CA
erica.snow@sri.com

Andrew E. Krumm
Center for Technology in Learning
SRI International
333 Ravenswood Ave
Menlo Park ,CA
andrew.krmm@sri.com

Timothy Podkul
Center for Technology in Learning
SRI International
333 Ravenswood Ave
Menlo Park, CA
timothy.podkul@sri.com

Mingyu Feng
Center for Technology in Learning
SRI International
333 Ravenswood Ave
Menlo Park, CA
mingyu.feng@sri.com

Alex J. Bowers
Teachers College
Columbia University
525 West 120th St.
New York, NY 10027
bowers@tc.edu

ABSTRACT
The current study employs transitional probabilities as a way to classify and trace students' interactions within an online learning system. Results revealed that students' interaction patterns within the system varied in relation to their performances on embedded assessments. The results and methodologies presented here are designed to provide practitioners with a starting place for how to extract information concerning how and why their students interact within an online environment.

Keywords
Blended Learning, Transitional Probabilities, Online Technology

1. INTRODUCTION
The use of blended learning techniques has become increasingly prevalent within high school classrooms [1]. One goal of blended learning is that information concerning students' behaviors and performance within various technologies can be used to inform instructional practice [2]. However, trace-level data from most technologies are often inaccessible or unusable for practitioners [3]. The current work aims to better understand what methodologies and tools are useful for helping practitioners make sense of *how* students interact with assessments and resources within online technologies. Using transitional probabilities we examined how 812 middle and high school students interacted with an online learning system (OLS) as part of their regular Math classroom instruction and how these behaviors varied as a function of students' performance within the system.

2. METHODS

2.1 Participants
The participants included 812 students from a large charter management organization (CMO) in the San Francisco Bay area. Over 60% of students who attend this CMO come from underserved populations (e.g., African American and Hispanic or Latino) and over 40% qualify for free or reduced priced lunches. The participating students regularly interact with the OLS as part of their Math curriculum.

2.2 Procedure, Measures, and Data Processing
Students interacted with the Math content on the OLS throughout the 2014-2015 school year. In the work presented here we examined how students interacted in one lesson for their Math curriculum, *Linear Equations*. During this lesson, students could freely choose to engage in a variety of activities at their own pace. These activities can be grouped into three categories that represent a different type of functionality within the system; these functionalities are *Post Assessments* (Linear Equation content gleaned from system resources), *Pre Assessments* (baseline measure of students' Linear Equation knowledge), and *Resources* (unique items –PDFs, videos, images- that provide Linear Equation content). These categories afforded the opportunity to trace students' choice of interactions within the system while also providing a means of surfacing reoccurring patterns of behavior that students exhibit throughout the school year. All interactions are logged within the system and provide valuable insight into *how* students interact with the OLS.

3. QUANTITATIVE METHODS
To examine variations in students' behavior patterns within the Linear Equation curriculum of the OLS, transitional probabilities were conducted. This analytical tool provides a means to provide teachers with a visualization of students' learning trajectories. This is particularly useful for practitioners interested in examining how closely students' choices followed the intended system curriculum. The following section provides a brief description and explanation of transitional probabilities and their application to the current data set.

3.1 Transitional Probabilities
Transitional probabilities were calculated using a statistical sequencing procedure established in D'Mello, Taylor, and Graesser (2007; [4]). This sequencing procedure is calculated using the formula $L[I_t \rightarrow X_{t+1}]$. In this formula, L is the likelihood function of the student's current choice in the system (I) at specific time point t, and X is their next interaction choice at the next time point (t+1). Thus, this sequencing procedure surfaces the probability of a student's interaction choice given their previous choice. For instance, if Zach chooses to take a Pre

Assessment, the above formula will be used to surface what choice Zach is most likely to choose next (e.g., another Pre Assessment, a Post Assessment, or a Resource). These probabilities were calculated for each of the 812 students, which resulted in a unique pattern of choices for each student. The results reported below address students' interactions with the Pre Assessment, Post Assessment, and Resources associated with Linear Equations content within a 9th grade Math course.

4. RESULTS

Overall, 812 students interacted with the Linear Equation content within the OLS system. Teachers recommended that students take the Pre Assessment, interact with system Resources, and then take the Post Assessment to measure changes in learned material. However, as this was a blended learning environment students were free to choose how they would spend their time and what features they would interact with. Using system log data, we classified students' interactions into one of three orthogonal categories (i.e., Post Assessments, Pre Assessments, and Resources). We classified students as passing if they scored at or above 80% and failing if the scored below 80%. To examine how students interacted with the system, we calculated the total frequency of students' interactions with each of these three categories. On average, students made 38 interactions within the system and spent the majority of their time interacting with Pre Assessments (53%), followed by taking Post Assessments (32%) and interacting with Resources (15%).

4.1 Interaction Transitions

The current work aimed to better understand how students' performance in Math 9 influenced their next interaction within the OLS. Figure 1, displays the conditional transition probabilities for students who passed a Post Assessment for Linear Equations. In this figure, there are three possible interactions, retrying a Post Assessment, transitioning to a Pre Assessment, or transitioning to a Resource. Students can also choose to move onto another topic. This analysis revealed that after students' passed a Post Assessment, .01% of the time they tried another Post Assessment, 1% of the time they took a Pre Assessment, and 17% of the time they interacted with a Resource. Most often after passing a Post Assessment, students left that content area to start another (72%).

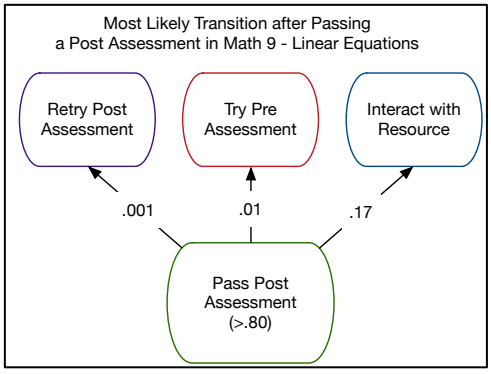


Figure 1. Conditional probabilities after passing Post Assessment.

Figure 2, displays the conditional transition probabilities if a student fails a Post Assessment for Linear Equations. Similar to Figure 2, there are three possible interactions along with students' choice to leave the curriculum. This analysis revealed that after students' failed a Post Assessment, 48% of the time they retook the Post Assessment, 43% of the time they took a Pre Assessment and 7% of the time they interacted with a Resource. Unlike students who passed a Post assessment (Figure 1), students who failed a Post Assessment were less likely to exit the curriculum (2%) and instead most often interacted with another form of assessment (Pre or Post).

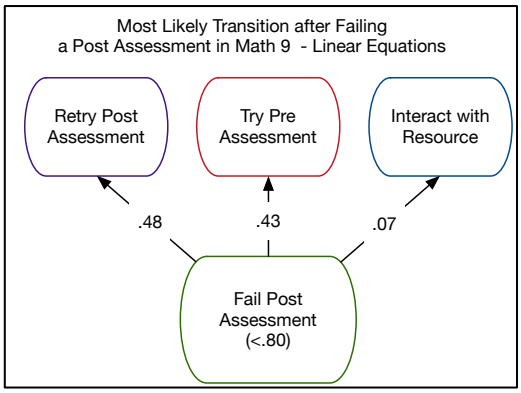


Figure 2. Conditional probabilities after failing a Post Assessment.

5. DISCUSSION

These exploratory findings are promising for both educational researchers and practitioners as they reveal how students' behavior patterns manifest and vary as a function of performance. The current work begins to shed light upon the nuanced ways in which students' interactions can be traced and classified within online environments. In the future, this work will be expanded to examine students' behavior patterns across multiple classrooms and courses. The goal will then be to examine how students' behaviors vary as a function of performance and domain. This information may prove useful to practitioners wishing to better understand how information extracted from technology can be used to inform instructional practices.

6. ACKNOWLEDGEMENTS

This research was supported by the National Science Foundation (DRL-1444621). The opinions expressed are those of the authors and do not necessarily represent views of the NSF.

7. REFERENCES

[1]. Stockwell, B. R., Stockwell, M. S., Cennamo, M., & Jiang, E. 2015. Blended learning improves science education. Cell, 162(5), 933-936.

[2]. Halverson, R., Grigg, J., Prichett, R., & Thomas, C. 2007. The new instructional leadership: Creating data-driven instructional systems in school. Journal of School Leadership, 17(2), 159.

[3]. Jacovina, M. E., Snow, E. L., Allen, L. K., Roscoe, R. D., Weston, J. L., Dai, J., & McNamara, D. S. 2015. How to visualize success: Presenting complex data in a writing strategy tutor. In O. C. Santos, J. G. Boticario, C. Romero, M. Pechenizkiy, A. Merceron, P. Mitros, J. M. Luna, C. Mihaescu, P. Moreno, A. Hershkovitz, S. Ventura, & M. Desmarais (Eds.), Proceedings of the 8th International Conference on Educational Data Mining (Madrid. Spain) EDM 2015 pp. 594-595.

[4]. D'Mello, S. K., Taylor, R., and Graesser, A. C. 2007. Monitoring affective trajectories during complex learning. In D. S. McNamara & J. G. Trafton (Eds.), Proceedings of the 29th Annual Meeting of the Cognitive Science Society (Nashville, Tennessee, August 1-4, 2007) Cognitive Science Society, 203-208.

A Platform for Integrating and Analyzing Data to Evaluate the Impacts of Educational Technologies

Daniel S. Stanhope
Lea(R)n, Inc.
Raleigh, NC 27603
daniel.stanhope@learntrials.com

Karl T. Rectanus
Lea(R)n, Inc.
Raleigh, NC 27603
karl@learntrials.com

ABSTRACT

Educational technology (edtech) products are ubiquitous in schools, but a paucity of research has evaluated their impact on education outcomes. Herein we describe a platform (i.e., LearnPlatform) that enables users to integrate and analyze data to rigorously evaluate the impacts of edtech. The platform also enables users to mine large and diverse datasets to identify patterns and trends in edtech usage and impact, and to build statistical models through predictive analytics that use multiple predictors to forecast future events, trends, and probabilities. Ultimately, educators and researchers can use LearnPlatform to generate evidence-based insights about edtech ecosystems within and across schools, districts, and states, which will improve the discovery, purchasing, and evaluation of edtech products in myriad educational contexts.

Keywords

Educational technology, efficacy, data, evaluation, education outcomes

1. INTRODUCTION

Educational technology (edtech) is increasingly pervasive. Each year, billions of dollars are spent and innumerous products are released. Despite immense resources invested, there has not been a standard system for monitoring and evaluating the use, quality, and efficacy of edtech products, leaving school leaders without access to critical data when making instructional, operational, and fiscal decisions. These decision makers need timely, reliable, evidence-based information on edtech interventions to know what to buy, how to support instruction and implementation, and how to improve student outcomes. Accordingly, Lea(R)n, Inc. worked with thousands of educators, state and district leaders, subject matter experts, and researchers to develop an online edtech management platform, called LearnPlatform, to help education organizations and institutions understand and manage which edtech products are best for their needs.

2. EDTECH MANAGEMENT PLATFORM

LearnPlatform is an edtech management platform that helps schools and districts understand which edtech products are best for their classrooms and students. To ensure valuable and trustworthy insights, the platform was built to support sound research methods and study designs[1] that enable systematic investigations within authentic educational contexts. The platform offers a research-based system for educators to understand, manage, and evaluate edtech products. Among other things, the platform allows users to (a) identify, catalogue, and monitor the products that are being used in their classrooms; (b) grade products on a valid and reliable rubric;[2] (c) connect with colleagues to share insights and ask questions; and, (d) conduct edtech evaluations that range from rapid-cycle pilots to randomized control studies (RCTs) to multi-product factorial studies. The analytics module of the platform, called LearnTrials IMPACT (*Integrating Metrics for Producing Analytics on Classroom Technology*), allows users to rapidly integrate disparate datasets and analyze those data to generate evidence-based insights on edtech interventions.

3. ANALYTICS MODULE

The platform's analytics module (LearnTrials IMPACT) has several noteworthy components. First, the platform maintains and continuously updates a relational database with over 4,000 edtech products that are available to educators (see Figure 1).[3]

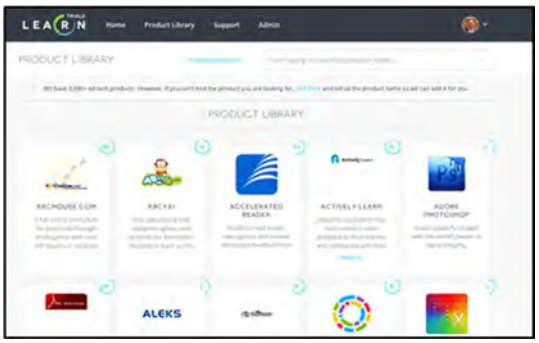


Figure 1. Screenshot of product library with product grades.

Second, a structured architecture allows educators to leverage useful features, including managing portfolios of products, sharing experiences with tools, asking colleagues questions, viewing products' grade reports, and comparing products side by side (see Figure 2 for example of an administrator view).

Third, capabilities of the platform allow districts to collect rapid feedback on the products they already use, launch evaluations of products, and analyze findings filtered by dozens of criteria (e.g., purpose of product use, frequency of use, student groups with which the product is used; see Figure 3).

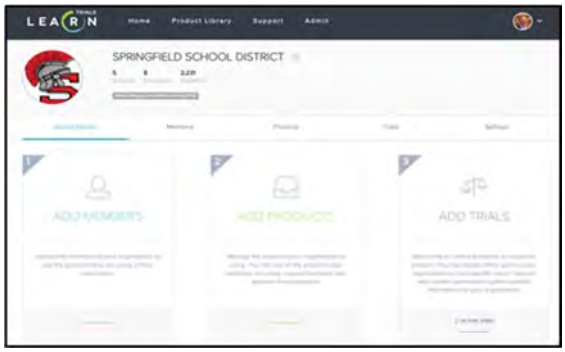


Figure 2. Administrator view of LearnPlatform.

Figure 3. Screenshot of functionality in the IMPACT layer.

Fourth, the platform aggregates educators' evaluations of products into interpretable and actionable recommendations about the product and its optimal use with various student populations. Finally, a data integration and automated analytics layer allows users to rapidly de-identify, upload, and analyze product usage (e.g., time on system, modules completed), student outcomes (e.g., achievement, motivation, engagement), and other data to produce dynamic reports and dashboards that inform instructional, operational, and budgetary decisions (see Figure 4 for example of Impact Analysis Report with simulated data and a fake product).

4. CASE STUDY

Schools, districts, and states across the US are using LearnPlatform. One of the nation's largest school districts leveraged LearnPlatform to conduct a controlled trial with a quasi-experimental design that generated insights for budgeting and implementation. In the efficacy trial, the district studied a widely used edtech product for elementary literacy. The sample included 18 schools who used the product (treatment group; $n_T > 8{,}000$) and 18 schools who did not use the product (control group; $n_C > 8000$). We tested for baseline equivalence on multiple measures, including demographics and prior achievement. We also applied statistical adjustments to control for variance attributable to extraneous factors and covariates. We first computed covariate-adjusted effect sizes to determine the extent to which the product exhibited an impact on the treatment versus the control, then conducted cluster analysis to identify student clusters of product usage and examined achievement for different clusters. Results were confirmed through a separate, blind analysis by the district's data and accountability office. Additional analysis of costs informed the district's purchasing and budgeting decisions.

Figure 4. Example of an impact report (fake product and school).

5. FUTURE DIRECTIONS

First, LearnPlatform will enable users to mine datasets to identify patterns and trends in edtech usage and impact, and to build statistical models through predictive analytics that forecast future events, trends, and probabilities. Second, once enough data are available, users will be able to leverage LearnPlatform to conduct meta-analyses to begin to elucidate conditional and contextual effects that may differentiate the efficacy of a given intervention based on factors that vary across schools, districts, or states. Ultimately, educators and researchers will use LearnPlatform to gain data-driven insights into edtech ecosystems across schools, districts, and states, and to improve discovery, purchasing, and evaluation of what works for educators and their organizations.

6. ACKNOWLEDGMENTS

Development of the IMPACT layer has received funding from organizations such as the Bill and Melinda Gates Foundation.

7. REFERENCES

[1] Lea(R)n, Inc. (2015, November 8). Grading EdTech: Our Rubric Effectively Differentiates Products. Retrieved from http://go.learntrials.com/rubric-research/

[2] Singer, N. (2016, January 17). Education Technology Graduates From the Classroom to the Boardroom. Retrieved from http://www.nytimes.com/2016/01/18/technology/education-technology-graduates-from-the-classroom-to-the-boardroom.html?_r=1

[3] Creswell, J. W. (2013). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches (4th ed.)*. Thousand Oaks, CA: SAGE Publications

Patterns of Usage from Educational Technology Products across America

Daniel S. Stanhope
Lea(R)n, Inc.
310 S. Harrington St.
Raleigh, NC 27603
daniel.stanhope@learntrials.com

Karl T. Rectanus
Lea(R)n, Inc.
310 S. Harrington St.
Raleigh, NC 27603
karl@learntrials.com

ABSTRACT

Educational technology (edtech) products are ubiquitous in schools, yet there is a dearth of research examining their use and efficacy. This leaves schools and districts without evidence to inform important decisions about edtech budgeting, instruction, impact, and implementation. We report results from a study that uncovered startling trends in edtech usage across multiple paid products and dozens of schools. Notably, 36.6% of purchased student licenses were never used. An additional 28.2% of the licenses were used negligibly, failing to meet a quarter of the fidelity goal set by the product companies or districts. Further, anecdotal evidence suggests school- and district-level leaders are unaware of these realities. This suggests a vast amount of resources are being unknowingly squandered or misallocated. Combined with analysis of how product usage impacts student achievement, these results demonstrate how schools and districts can utilize data to understand and manage their edtech ecosystems while improving critical edtech decisions.

Keywords

Educational technology, efficacy, fidelity, evaluation, education

1. INTRODUCTION

Educational technology (edtech) presents both opportunities and challenges for educators and their organizations. Challenges include allocating resources appropriately, implementing products with fidelity, and ensuring product efficacy. Unfortunately, these challenges have been exacerbated because heretofore districts have not had systems or methods for collecting, comparing, and analyzing disparate data sources in a way that informs budgetary or instructional decisions. To address that lack of evidence, schools and districts across the nation have been using LearnTrials—a module on the LearnPlatform—to measure an integrated system of data and variables, enabling them to generate key insights and rapidly make informed decisions. In this paper, we report a specific set of early findings from a synthesis of systematic research focusing on edtech usage patterns, and we discuss the implications for implementation, impact, and budgeting.

More than $8 billion (PreK-12 alone) are spent annually on edtech products in the US with the goal to improve important education outcomes.[1] Both producers and consumers of edtech products worry about using them with fidelity—that is, ensuring students receive the "recommended dosage" to achieve the intended outcomes. Most agree that implementation and its impacts on budget and achievement are interrelated and worthy of treatment as a system; however, limited research has examined fidelity of edtech usage. This has led dozens of schools and districts to use LearnTrials to conduct rapid, cost-effective evaluation of multiple products, analyzing both edtech usage and efficacy.

2. METHODS

2.1 SAMPLE

The sample for this study is 49 K-12 schools in multiple districts and states. Overall, the sample included over 17,000 students from a diverse set of schools. For each school, we examined data on product usage collected during the 2014-2015 academic year. Specifically, we tracked the extent to which students used their licenses for six well-known digital math and literacy tools. Each of these products was well-established in the marketplace, used for primary instruction (rather than supplemental), and ranged in price from $16 to over $100 per student, per year.

2.2 ANALYSIS

The main analysis for this study involved descriptive statistics on the extent to which students used their product licenses. Each of the six products prescribe a specific amount of student usage, often called the recommended dosage. In other words, these products have predetermined metrics for usage goals (e.g., time logged in, progress through syllabus, number of lessons passed) intended to promote marketed outcomes. Based on these measures, we analyzed the extent to which students met certain expectations. Specifically, we examined whether students (a) never used the product, (b) used the product but failed to meet even 25% of the goal, (c) met 25% of the usage goal, (d) met 50% of the usage goal, or (e) fully met the usage goal.

3. RESULTS

We found consistent patterns of usage across the schools and across the products. The main finding: 36.6% of purchased product licenses were never activated. An additional 28.2% of students activated their license, but did not use the product enough to meet even 25% of the established goal. Thus, approximately 64.8% of students exhibited zero or trivial use. Moreover, only 5.2% of students actually received the full recommended dosage (Figure 1; see Figure 2 for a breakdown of

use by product). In summary, schools are paying significant amounts of money for products that students are not using.

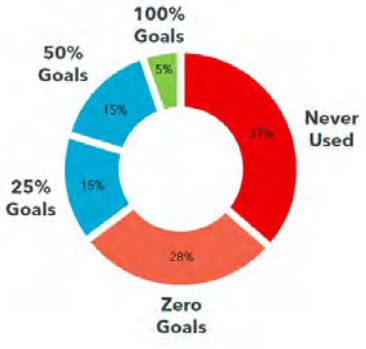


Figure 1. Percent of paid product licenses meeting dosage goals.

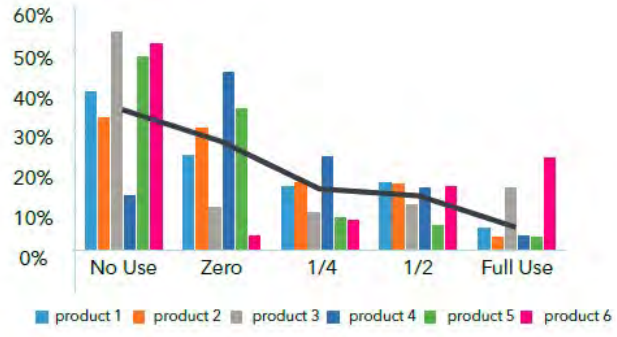


Figure 2. Paid product licenses meeting dosage goals by product. (Product names undisclosed for sake of anonymity.)

4. DISCUSSION

To be clear, the startling lack of product usage across schools is not an indictment of edtech products or the schools that use them—classroom technologies are valuable, and have the potential to amplify learning. While these are early findings, they have numerous implications for schools and districts.

Implementing learning technologies in schools and districts presents opportunities and challenges. One way to maximize the former and minimize the latter is understanding important contextual factors. Recognizing the specific factors that impact use within local contexts can uncover opportunities for growth. Structured pilots, rapid feedback cycles, and scaled roll-outs do not have to be cumbersome. Leveraging data-rich product pilots can address common challenges. By using research-backed, standardized edtech management systems in their local contexts, districts can lower opportunity costs, reduce negative impacts on teaching and learning, and mitigate political consequences of "all-in, all-at-once" implementations.

Understanding product efficacy—the extent to which a product impacts intended educational outcomes—is important. The U.S. Dept. of Education, the Bill and Melinda Gates Foundation, and others have recently invested in rigorous and realistic evaluation of products at every stage. If students do not use a product, they cannot capitalize on its potential benefits. Discovering that edtech products are consistently underused (or never used) is a first step. Providing schools and districts insights into situational variables (e.g., student characteristics, school types, demographics, or pedagogical styles) would help educators and product companies understand the contexts in which products have positive, negative, or negligible impact. Our research has shown times when minimal (and even significant) usage had deleterious effects on student achievement. In other cases, specific student groups using certain edtech products saw greater gains than did their peers. Delivering context-specific insights that are based on statistical analysis via timely, easy-to-understand dashboards and reports help schools and districts identify the best tools for their situations and instructional needs.

A final implication is the obvious impact on budget. If we extrapolate the findings reported herein, it is likely that last year schools spent nearly $3 billion on product licenses that were never activated (37% of the $8 billion spent across U.S. schools). However, edtech purchasing decisions do not exist in a vacuum; rather, they are richly contextualized and made based on budgetary constraints, merit of competing products, politics, and precedent. Challenges also include current business models, lack of pricing transparency, and unknown usage data. Furthermore, edtech purchasing has decentralized rapidly, meaning individual educators and schools are making more decisions, which creates organizational challenges for district and state leaders.

Educators and their organizations need a systematic approach for gathering evidence,[2] and for rapidly understanding organization-wide product usage and efficacy. Analysis of local data as well as analysis of large-scale databases can greatly enhance our ability to evaluate edtech phenomena.[3] Then, implementing edtech management systems, service level agreements, and performance contracts (based on successful usage or other measurable milestones) are not only possible, but also capable of improving instruction, finances, and educational outcomes.

The consistent patterns of usage—specifically the limited use of paid licenses—across edtech products in education environments offers a massive opportunity to improve a complex system. Until recently, edtech decisions lacked a systematic approach for measuring and collecting evidence on the most important variables. However, dozens of schools and districts are using the edtech management LearnPlatform and its LearnTrials module to analyze their edtech ecosystems in unbiased and rapid ways, so they can make evidence-based decisions that enhance the fidelity of implementation, boost product impact on student achievement, and maximize resources (e.g., time and money).[4]

5. REFERENCES

[1] Richards, J. & Stebbins, L. (2014). 2014 U.S. Education Technology Industry Market: PreK-12. Washington, D.C.: Software & Information Industry Association.

[2] Coburn, C. E., Honig, M. I., & Stein, M. K. (2009). What's the evidence on districts' use of evidence? In J. D. Bransford, D. J. Stipek, N. J. Vye, L. M. Gomez, & D. Lam (Eds.), The role of research in educational improvement (pp. 67-87). Cambridge, MA: Harvard Education Press.

[3] Penuel, W. R., & Means, B. (2011). Using large-scale databases in evaluation: Advances, opportunities, and challenges. *American Journal of Evaluation*, 32, 118-133.

[4] Johnson, K. (2016, March 15). Resources to Help You Choose the Digital Tools Your Classroom Needs. Retrieved from https://www.edsurge.com/news/2016-03-15-resources-to-help-you-choose-the-digital-tools-your-classroom-needs

Learning curves versus problem difficulty: an analysis of the Knowledge Component picture for a given context

Brett van de Sande
Pearson Education
brett.vandesande@pearson.com

ABSTRACT

The Knowledge Component (KC) picture of learning has proven useful for constructing models of student learning in a number of subject areas. However, it is still unclear how well this picture generalizes to other contexts and subject areas. A corpus of 62,000 exercises for 10 textbooks on the Mastering platform has been tagged by content experts. In this report, I introduce a strategy for investigating the importance of a given set of KCs in describing student performance as the students solve problems. The strategy is to see how much of the student's performance on an exercise is explained by the associated KC and how much it is predicted by a problem-specific difficulty parameter. To do this, I introduce a model that is a combination of the Rasch model and the learning curves from the KC picture. For this corpus and set of KC tags, a rather striking picture emerges: problem difficulty accounts for most of the student behavior while KC learning accounts for only a small portion of the student behavior. I hypothesize that these KC tags do not accurately capture the skills students are using while doing their homework.

Author Keywords

Learning Curves, Knowledge Components

ACM Classification Keywords

I.2.6 Learning: Knowledge acquisition

Knowledge components (KCs) are bits of information needed to solve a problem [5, 2]. KCs generally have some sort of pre-requisite relations. However, aside from prerequisites, a KC can, by definition, be mastered independently from other KCs. This definition assumes that KCs are *context independent*. That is, the student's ability to apply that KC correctly or quickly does not depend on the particular problem the student is solving or the other KCs needed to solve that problem.

Since KCs are *defined* to have these properties, then it remains to be seen whether a given set of KC labels for a particular curriculum provides a useful description of skill acquisition. Much of the pioneering work on KCs focused on middle school math [4]. It is unclear whether this picture extends to the corpus examined here.

Table 1. Some Knowledge Components for Chapter 32 of "University Physics" by Young, Freedman, and Lewis [6].

1	Relationship between speed of electromagnetic (EM) waves, wavelength and frequency
2	Writing Maxwell's equations for free space. Using Faraday's Law.
3	Direction of propagation of an electromagnetic wave

One way to determine how well the KC picture is working is to examine the associated learning curves. If the curves increase/decrease more-or-less monotonically (depending on the measure of competence) then the KC picture is working. A smooth learning curve implies that the associated KCs account for most of the student performance on a problem while other aspects of the problem are less important.

A corpus of over 62,000 exercises on the Mastering platform has been tagged by content experts. This corpus covers homework exercises for 10 college-level textbooks in anatomy and physiology, biology, organic chemistry, general chemistry, and physics. An typical set of KCs is shown in Table 1. On average, there are about a dozen KCs per chapter.

We examined log data from problems solved on the Mastering platform during the Spring of 2014. We selected students whose coursework spanned more than 25 days and who were enrolled in a course containing more than 50 students.

Before we address the main question of the validity of the KC picture for this corpus, we mention some general properties of the log data. The learning curves (see Fig. 1) are expressed in terms of "difficulty" which is defined to be minus the logistic of the probability of "correct on first try."

The mean number of opportunities to practice a given KC is 3.84, averaged over students and KCs. So, students have very few opportunities to practice a given KC.

Also, the number of students practicing a KC usually decreases rapidly with increasing opportunity number t. This can result in a selection bias, since the population is changing with t. Thus, to produce a learning curve for a given KC, we rank the students by the total number of opportunities for that KC and take the uppermost portion as our student population. An example learning curve is shown in Fig. 1. In general, we

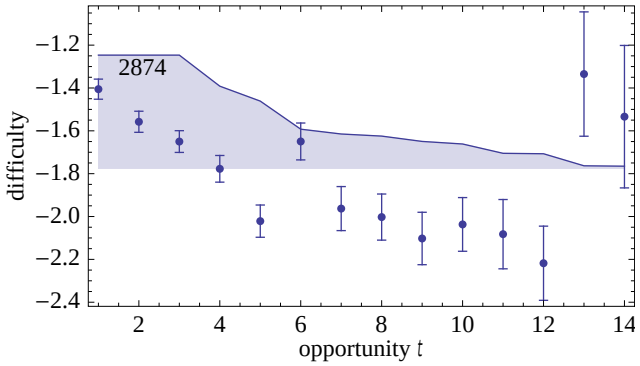


Figure 1. Learning curves for the first KC listed in Table 1. Difficulty should decrease as students learn. The shaded region represents the relative number of students who completed that opportunity and the number in the upper left corner is the initial number of students.

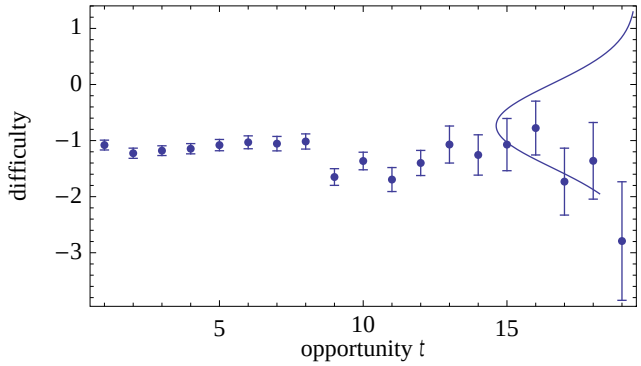


Figure 2. KC difficulties $\zeta_{k,t}$ versus opportunity number t from the Rasch/KC model applied to student log data for the first KC in Table 1. The curve on the right is a gaussian that represents the distribution of problem difficulties for the exercises labeled with the associated KC.

find that learning curves are not monotonically decreasing. In fact most do not even show a decreasing trend.

There must be important aspects of the exercises that are not captured by these KCs. Thus, we introduce problem difficulty β_p to capture the aspects of a problem not explained by the KCs. This leads us to introduce the Rasch/KC model: a hybrid of the Rasch model [3], and the learning curve picture.

If $P_{s,p}$ is the probability that student s gets problem p correct, then we define $P_{s,p}$ by the logistic equation:

$$\text{logit}\,(P_{s,p}) = \theta_s - \beta_p - \sum_{(k,t)\in\mathcal{T}_{s,p}} \zeta_{k,t} \qquad (1)$$

where θ_s is the skill of student s, β_p is the difficulty of exercise p, and $\zeta_{k,t}$ is the difficulty of applying KC k on opportunity t. $\mathcal{T}_{s,p}$ is the set of KC, opportunity pairs where $(k,t) \in \mathcal{T}_{s,p}$ means that problem p is opportunity t for student s to apply KC k. The log-likelihood for a set of students and problems to obtain a particular set of outcomes is

$$\log(\mathcal{L}) = \sum_{s,p\in\mathcal{C}_s} \log(P_{s,p}) + \sum_{s,p\in\mathcal{I}_s} \log(1-P_{s,p}) + \quad (2)$$

where $\mathcal{C}_s$/$\mathcal{I}_s$ is the set of problems s got correct/incorrect.

If we drop $\zeta_{k,t}$, then we obtain the usual Rasch model. Likewise, if we drop θ_s and β_p and fit the resulting model to student data, a plot of $\zeta_{k,t}$ versus opportunity t will yield the conventional learning curve for KC k; this is precisely what we have plotted in Fig. 1. This model is similar to the Additive Factors Models (AFM) [1] except that AFM restricts $\zeta_{k,t}$ to be linear in t.

We can apply this model to student log data associated with the KCs listed in Table 1. We find that both student skills $\{\theta_s\}$ and problem difficulties $\{\beta_p\}$ are Gaussian distributed with standard deviations of 1.02 and 1.15, respectively.

Looking at the KC difficulties $\zeta_{k,t}$ in Fig. 2 we see that the difficulties vary little with opportunity number. We also, see that the associated problem difficulties, represented by the Gaussian distribution on the right, vary significantly more than the KC difficulties. The same qualitative behavior is seen for all KCs we have analyzed. We conclude that, for this corpus and KC labeling, problem difficulty is much more important than KC mastery when predicting student performance on an exercise.

If we look at the KCs, see Table 1, we see that they represent content knowledge rather than more abstract problem solving skills. It may be that the students have already learned the content knowledge in lecture or reading and, during their homework, they are really learning how to apply that content knowledge to various physical situations. If this is the case, it may be more appropriate to label problems with labels that are more oriented towards problem-solving skills, like "given description of situation, determine that one should relate velocity, frequency, and wavelength." Also, it may mean that one can explain student performance with just a few KCs like "solve physics word problem" or "solve problem with kinematics graphs."

REFERENCES

1. Chi, M., Koedinger, K., Gordon, G., Jordan, P., and VanLehn, K. Instructional Factors Analysis: A Cognitive Model For Multiple Instructional Interventions. In *Proceedings of the 4th International Conference on Educational Data Mining* (Eindhoven, the Netherlands, June 2011).
2. Koedinger, K. R., Corbett, A. T., and Perfetti, C. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Sci. 36*, 5 (2012), 757–798.
3. Rasch, G. *Probabilistic Models for Some Intelligence and Attainment Tests.* MESA Press, Chicago, 1993.
4. Ritter, S., Anderson, J. R., Koedinger, K. R., and Corbett, A. Cognitive Tutor: Applied research in mathematics education. *Psychon. B. Rev. 14*, 2 (Apr. 2007), 249–255.
5. VanLehn, K. The Behavior of Tutoring Systems. *Int. J. Artif. Intell. Ed. 16*, 3 (Jan. 2006), 227–265.
6. Young, H. D., Freedman, R. A., and Ford, A. L. *University Physics with Modern Physics*, 13 edition ed. Addison-Wesley, Boston, Jan. 2011.

Validating Automated Triggers and Notifications @ Scale in Blackboard Learn

John Whitmer, Ed.D.
Blackboard, Inc.
58 Maiden Lane, 5th floor
San Francisco, CA 94108
+1(530)554-1528
john.whitmer@blackboard.com

Sasha Dietrichson, Ph.D.
Blackboard, Inc.
1111 19th Street, NW
Washington, DC 20036
+1(800) 424-9299
sasha.dietrichson@blackboard.com

Bryan O'Haver
Blackboard, Inc.
190 W. Ostend Street, Suite 205
Baltimore, MD 21230
+1(800) 424-9299
bryan.ohaver@blackboard.com

ABSTRACT

Prior research on individual courses has demonstrated a significant relationship between use of the Learning Management System (LMS) and student course grade. Blackboard has created rule-based algorithms in a new LMS interface to notify students and faculty of students who may be at risk based on relative activity and grades received, and recognize positive behavior and grade achievement. This research project investigated the relationships underlying these algorithms against a large data set of LMS activity (1.2M student course weeks, 34,519 courses, 788 institutions). Findings included a small effect size in the relationship between time spent in the LMS and student grade; however, a small set of courses had a strong relationship that merits further research and consideration.

Keywords

Learning Analytics, Student Persistence, Student Retention, Higher Education, Learning Management Systems, LMS

1. INTRODUCTION

Multiple research studies on individual courses have found a significant relationship between use of the LMS and student grade [8, 7, 2, 3, 9, 10]. The value of LMS data in these courses has been larger than what is found in conventional demographic or academic experience variables in explaining variation in course grades. However, when analysis is expanded to all courses at an institution, several studies have found no relationship or an extremely small effect size in this relationship [1, 5, 4]. Does Learning Analytics only apply to only a small number of courses, or is it broadly applicable? What is the magnitude of this relationship, and is sufficiently large to include algorithms based on this relationship as a core functionality in academic technology platforms?

This question is of great practical significance for academic technology providers. Analytics functionality has typically been provided through custom data warehouses and analytics tools that include multiple data sources and systems, with custom integrations and algorithms. While useful and with accuracy that can be proven, these applications require significant resources to create and maintain, whether procured from a vendor or built in-house. They also require significant time to implement and deploy.

As part of Blackboard's new "Ultra" LMS course interface, rule-based triggers and notifications were created. For example, these rules would analyze course use and send the student and instructor a notification if a student's LMS activity dropped more than 10% from one week to the next. In addition to alerts of potentially at-risk students, positive encouragement alerts were also created to recognize outstanding achievement relative to self and others in the same course.

The rules were created based in prior research findings and an initial small data sample. However, additional validation with a larger data sample was required to ensure that the rules were meaningful predictors of student grade. This poster presents findings from this research on the question of accuracy and draws broader conclusions about the potential utility and generalizability of LMS activity data.

2. DATA SET AND ANALYSIS

The data analyzed for this project was sampled from log data recorded by Blackboard Learn. These logs were transformed into normalized data sets, and calculations made to estimate duration of time spent in the LMS by calculating the difference between start end end times for sessions. The data was aggregated at the institution-course-week-user level (e.g. one row per user per week per course per institution). The data sample included a complete set of students active for each course week, but did not include all weeks for each course. Each row also contained final course duration and final grade. A z-score of duration was calculated to provide a course-specific measure of student activity.

Given the importance of analyzing grade triggers and the relationship between activity and grade, only course-weeks with a graded entry for that week were included in the sample. Further, students with no activity have no logs and are therefore missing. This biases the sample toward courses making more intensive use of the LMS than a random sample.

Exploratory data analysis revealed a large number of rows with invalid grades and duration. The data was filtered to include courses with valid data and a potential for instructional use, namely: grade range between 0% and 120%, a minimum of 60 average minutes in the course, and a maximum of 5,040 minutes in the course per week, and enrollment more than 10 student and less than 500 students.

The final data set analyzed had the following profile:

Table 1. Data Set Characteristics

Records	Courses	Institutions
1.2M	34,591	788

Exploratory data analysis and distributions were conducted to ensure that the data was normally distributed and ensure other assumptions required for linear regression analysis were met. A

linear regression of final course grade on course duration and a logistic regression of course pass/fail on duration was run. Next, a separate linear regression was run for each course.

3. FINDINGS

As indicated in the scatterplot in Figure 1, there was a significant relationship between duration and grade. However, the effect size was extremely small (adjusted R^2=0.01537). Further, most of this effect was created by the intercept value; the coefficient for duration was 5.74e-04. Converted into practical effect, this coefficient indicates that for each additional hour spent in the LMS, students would gain 0.034% in their final course grade. Using course-relative measures of duration (e.g. z-scores by course) only increased the effect slightly (R^2=0.017). Logistic regression led to similar results.

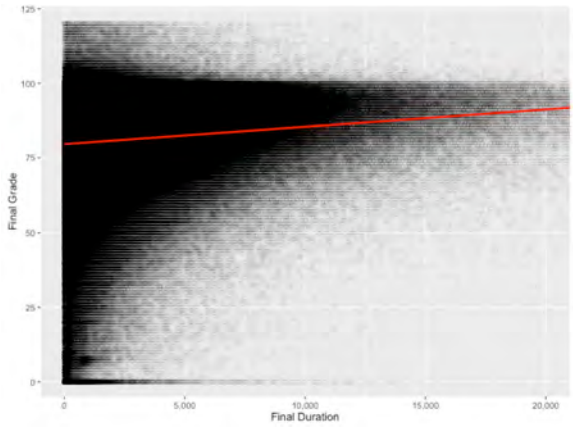


Figure 1. Duration vs. Grade across all Courses

When this regression was re-run at the course level, a high variation in this effect size was found. There were 7,648 (22%) courses with $p < 0.05$; the distribution in effect size is plotted below. Although skewed toward low values, there are a substantial number of courses with low to moderate effect sizes.

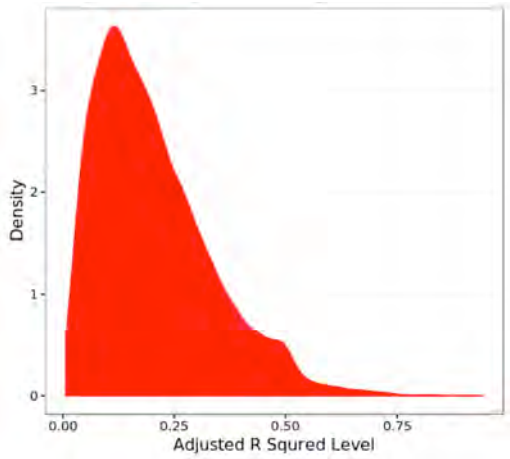


Figure 2. Adjusted R^2 Levels for Courses with Significant Duration vs. Grade Regressions

Initial data subsetting by available criteria (e.g. enrollment size, institution, average activity) did not identify a factor strongly related to this difference in effect.

4. IMPLICATIONS

These findings indicate that while rule-based triggers may not be predictive of student course achievement for all LMS courses, they are predictive for a substantial number of courses. Given known variability in how the LMS is used for instruction, these results provide an encouraging indication of potential value in this data. However, the reasons for this strong relationship among some courses and not among others is an important area for further research. We anticipate investigating issues in course design and early participation as identifiers of higher effect size.

As a result of this research, multiple modifications to the existing triggers in Blackboard Ultra have been made to refine and reduce the number of notifications sent. Further, a new configuration setting will be provided to disable these algorithms by course.

5. REFERENCES

[1] Campbell, J. P. (2007). Utilizing student data within the course management system to determine undergraduate student academic success: An exploratory study. (Educational Studies Ph.D.).

[2] Dawson, S., & McWilliam, E. (2008). Investigating the application of IT generated data as an indicator of learning and teaching performance (pp. 45). ASCILITE 2008, Melbourne.

[3] Fritz, J. (2011). Classroom walls that talk: Using online course activity data of successful students to raise self-awareness of underperforming peers. The Internet and Higher Education, 14(2), 89-97. doi: 10.1016/j.iheduc.2010.07.007

[4] Gašević, D., Dawson, S., Rogers, T., & Gasevic, D. (2016). Learning analytics should not promote one size fits all: The effects of instructional conditions in predicting academic success. The Internet and Higher Education, 28(January 2016), 68-84. doi: http://dx.doi.org/10.1016/j.iheduc.2015.10.002

[5] Lauria, E. J. M. B., Joshua. (2015, October 29-30, 2015). Mining Sakai to Measure Student Performance: Opportunities and Challenges in Academic Analytics. Paper presented at the European Conference on e-learning, Hatsfield, UK.

[6] Macfadyen, L. P., & Dawson, S. (2010). Mining LMS data to develop an "early warning system" for educators: A Proof of Concept. Computers & Education(54), 11.

[7] Morris, L. V., Finnegan, C., & Wu, S.-S. (2005). Tracking student behavior, persistence, and achievement in online courses. The Internet and Higher Education, 8(3), 221-231. doi: 10.1016/j.iheduc.2005.06.009

[8] Rafaeli, S., & Ravid, G. (1997). OnLine, Web Based Learning Environment for an Information Systems course: Access logs, Linearity and Performance. Paper presented at the ISECON 1997, Orlando, FL.

[9] Ryabov, I. (2012). The Effect of Time Online on Grades in Online Sociology Courses. MERLOT Journal of Online Learning and Teaching, 8(1).

[10] Whitmer, J., Fernandes, K., & Allen, B. (2012). Analytics in Progress: Technology Use, Student Characteristics, and Student Achievement. EDUCAUSE Review Online (July 2012).

Discovering 'Tough Love' Interventions Despite Dropout

Joseph Jay Williams
Harvard University
125 Mt Auburn St
Cambridge, MA 02138
joseph_jay_williams@harvard.edu

Anthony Botelho
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609 USA
abotelho@wpi.edu

Adam Sales
University of Texas
Statistics
Austin, TX 78712
asales@utexas.edu

Neil Heffernan
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609 USA
nth@wpi.edu

Charles Lang
New York University
239 Greene St
New York, NY, 10003
charles.lang@nyu.edu

ABSTRACT

This paper reports an application to educational intervention of Principal Stratification, a statistical method for estimating the effect of a treatment even when there are different rates of dropout in experimental and control conditions. We consider the potential value for using principal stratification to identify "Tough Love Interventions" – interventions that have a large effect but also increase the propensity of students to drop out. This method allowed us to generate an estimate of the treatment effect in an RCT without the selection bias induced by differential attrition by restricting analysis to just the inferred "stratum" of students who would not drop out in either condition. This paper provides a case study of how to appropriate the method of principal stratification from statistics and medical research fields to educational data mining, where it has been largely absent despite increasing relevance to online learning.

Keywords

principal stratification; selection bias; statistics; attrition; noncompliance; randomized controlled trial; experiment; online education

1. INTRODUCTION

A persistent problem in interpreting randomized experimental comparisons in learning environments is that the frequency of student dropout may vary between conditions . This is known as *differential attrition*, and causes problems with statistical inference [3] regarding the magnitude and direction of differences between treatment and control conditions. In cases where student completion is the metric of interest, such differences in condition are easily measured by the number of students to complete each; a problem arises, however, when performance is the metric of interest, as if less students drop out of one condition than the other, it is over-represented in the analysis causing unreliable results.

Differential attrition can mask the existence of what we label **"tough love"** interventions (TLIs). A TLI describes an intervention which introduces a treatment condition with features that cause some students to drop out, but has beneficial effects for students who persist. It is important to know *how much* such interventions impact a potential outcome in order to perform a cost-benefit comparison against the dropout rate. We believe that principal stratification is one tool that can be used to measure the effect of conditions in the presence of differential attrition and help identify TLIs.

2. ILLUSTRATIVE EXPERIMENT: IMPACT OF QUESTIONS ABOUT CONFIDENCE

In the preliminary data presented here, we consider a randomized controlled experiment (RCE) conducted within ASSISTments, a K-12 online and blended learning platform, reported in EDM 2015 [4]. Students were randomly assigned to either a condition of Treatment, where students were asked about their confidence in solving problems, or Control, where students were asked about technology usage. The data set used for analysis consists of 712 12-14 year olds in the eighth grade of a school district in the North East of the United States with 5,861 log records collected while students were solving math problems. The goal here is to estimate how the conditions differ in their impact on Mastery Speed, the number of problems needed to reach three consecutive correct responses indicating a sufficient level of understanding. It is important to note that a lower value in this metric indicates better performance.

3. ANALYTIC STRATEGY

Principal stratification [2, 5] is an approach to modeling causal effects for a subset of subjects defined subsequently to treatment assignment. For instance, it applies when issues of noncompliance, censoring-by-death, and surrogate outcomes within conditions have occurred. It uses two models, labeled here as the *Attrition* and *Outcome* models, to first stratify students and then estimate effects on a single stratum. Our

Attition model identifies four strata based on a student's likelihood to attrite: 1) **AA or Always Attriters:** Students who drop out regardless of condition. 2) **AC**: Students who complete if assigned to Treatment but drop out if assigned to control group. 3) **CA:** Students who only complete if assigned to Control. 4) **CC or "Never-Attriters":** Students who always complete regardless of condition; this is the stratum of interest for our work here, as it is the only group for which a treatment effect is well-defined.

True stratum membership is never observed, but must be inferred by the Attrition model using observed covariates, for which this work uses only the student's prior percent correctness labeled as acc_i. As attrition for one condition is known for each student, only the likelihood that the student would complete the opposing condition is inferred as seen in the following equations:

$$logit(Pr(completes_{i,ctrl} = 1)) = \alpha_{ctrl} + \beta_{ctrl} * acc_i$$

$$logit(Pr(completes_{i,treat} = 1)) = \alpha_{treat} + \beta_{treat} * acc_i$$

The Outcome model then observes only students placed in to the "Never-Attriter" stratum to estimate treatment effects. The equation used here utilizes the same covariates as the Attrition model with the addition of a dichotomized value of condition and a class-level variance term:

$$masteryspeed_i = \beta_{0s} + \beta_{1s} * acc_i + \beta_2 * cond_i + \sigma_i$$

The model parameters were estimated with Markov Chain Monte Carlo (MCMC) using four chains over 16000 iterations of which the first 8000 are omitted as a burn-in period allowing for convergence. The *Rhat* value shown in Table 1 reflects the degree of convergence of the Markov Chains, with the values near 1 indicating proper convergence. The results of the analysis are also seen in that table, and indicate that a TLI is not found as the effects of condition are not significant, falling within the confidence interval.

	mean	sd	0.95 CI	Rhat
Constant	1.78	0.13	(1.52,2.04)	1
Prior_Percent_Correct	-0.14	0.18	(-0.49,0.21)	1
Treatment	0.02	0.05	(-0.08,0.11)	1

	mean	sd	0.95 CI	Rhat
Constant	2.95	0.31	(2.34,3.55)	1
Prior_Percent_Correct	-1.33	0.39	(-2.09,-0.56)	1
Treatment	0.02	0.06	(-0.1,0.14)	1

Table 1: Typical Analysis: Coefficients for outcome model that predicts Mastery Speed based on Condition and Prior Accuracy, without using principal stratification (top) versus those coefficients using principal stratification (bottom).

4. SIMULATION STUDY

As no significance was found for coefficients in either case, a further comparison of principal stratification to traditional methods was conducted to verify principal stratification is beneficial in identifying such interventions when ground truth is known. The data generating model was designed to capture a tough-love intervention in which reliable difference could be found between conditions for students who would never drop out. For each simulated student, we assumed two latent/unobserved variables, intended to capture notions of *Grit* and *Ability*. There were two observed covariates, *prior percent complete*, which was a function of grit, and *prior percent correct*, which was a function of ability. The Outcome Variable (which might correspond to a post-homework quiz score) was a continuous variable that was a linear function of Ability.

A similar methodology to that described on the non-simulated dataset was then conducted. The coefficient for condition gave us a treatment effect for the never-attritor stratum. For comparison, we also conducted a Typical Analysis that estimated a treatment effect using ordinary least squares regression on all the data *without* using principal stratification and after 500 runs of the simulation, the 95% confidence interval from OLS included the average treatment effect for the never-attritors 62% of the time. In contrast, the principal stratification credible intervals were more efficient/reliable, including the true treatment effect 91% of the time.

5. CONCLUSION

This paper presented an explanation and case study application of principal stratification, to illustrate its potential as a method for analyzing randomized experiments and interventions in digital learning environments. One example from our analysis was identifying "Tough Love Interventions", but differential attrition pose a wide range of challenges to analyzing data from experiments, especially as learners gain flexibility in online environments such as Massive Open Online Courses (MOOCs). This makes the reliable analysis of experiments with variable dropout and attrition of increasing importance to the educational data mining community.

6. ACKNOWLEDGMENTS

This work is partially supported by the United States National Science Foundation Grant #DRL-1420374 to the RAND Corporation.

7. REFERENCES

[1] J. D. Angrist, G. W. Imbens, and D. B. Rubin. Identification of causal effects using instrumental variables. *Journal of the American statistical Association*, 91(434):444–455, 1996.

[2] C. E. Frangakis and D. B. Rubin. Principal stratification in causal inference. *Biometrics*, 58(1):21–29, 2002.

[3] J. Heckman, N. Hohmann, J. Smith, and M. Khoo. Substitution and dropout bias in social experiments: A study of an influential social experiment. *Quarterly Journal of Economics*, pages 651–694, 2000.

[4] C. Lang, N. Heffernan, K. Ostrow, and Y. Wang. The impact of incorporating student confidence items into an intelligent tutor: A randomized controlled trial. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[5] A. C. Sales and J. F. Pane. Exploring causal mechanisms in a randomized effectiveness trial of the cognitive tutor. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

Stimulating collaborative activity in online social learning environments with Markov decision processes

Matthew Yee-King
Computing, Goldsmiths
University of London
m.yee-king@gold.ac.uk

Mark d'Inverno
Computing, Goldsmiths
University of London
dinverno@gold.ac.uk

ABSTRACT

Our work is motivated by a belief that social learning, where a community of students interact with eachother to co-create and share knowledge, is key to our students developing 21st century skills. However, convincing students to engage in and value this kind of activity is challenging. In this paper, we employ a technique from AI research called a Markov Decision Process (MDP) to model social learning activity then to suggest interventions that might increase the activity. We describe the model and its validation in simulation and draw conclusions about the effectiveness of this approach in general. The main contributions of the paper is to (i) show how it is possible to model education data as an MDP (ii) show that the resulting decision policy succeeds in guiding the system towards goal states in simulation.

Keywords

Social learning; Education system modelling, MDP, MOOC

Categories and Subject Descriptors

K.3.1 [**Collaborative learning**]: K.3.2 Computer science education G.3 Markov processes

1. INTRODUCTION

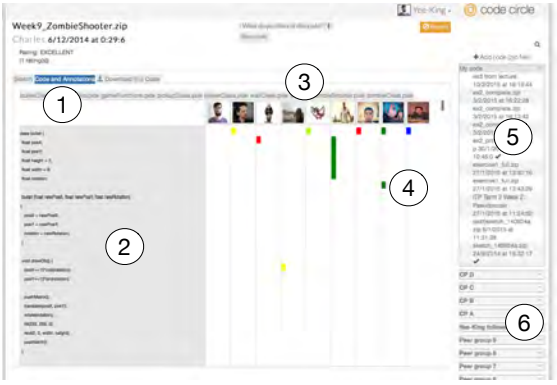
In this paper, we use a Markov Decision Process (MDP) to model social learning activity in terms of content consumption and content creation. This allows us to derive an 'action policy' which can potentially inform tutors and students what type of content to create and when to create it in order to maximise the levels of consumption of content in a social learning system. MDPs [2] are a commonly used method for sequential decision making under uncertainty, and they have been used in education technology e.g. [1]. The work presented here represents a novel application of MDP in a social learning context[1].

[1]A full version of the paper can be found at http://dx.doi.org/10.13140/RG.2.1.3592.0242

1.1 The case study and data set

The data used for the analysis presented here was collected during a 10 week case study involving 174 students on an introductory undergraduate programming course who were learning how to program using the Processing IDE. The students were using our social learning environment [3], as shown in Figure 1, which allow in-browser execution of programs as well as sharing, commenting and replying to comments on specific sections of code.

Figure 1: **The code discussion UI. 1) mode buttons: view running program, view code, download code, 2) the code viewer 3) the people who have commented on this code 4) a comment about a section of the code 5) my uploaded content 6) my communities.**



2. THE MODEL

MDP problems are formulated in terms of states, actions, state transitions, reward functions and action policies. The action policy dictates what is the best action to take in a given state in order to maximise future reward, where reward is defined in terms of the value of each state.

We begin by slicing the dataset into time windows and counting the number of activity types per window, split into content consumption and content creation activities. We define state as a 5 dimensional vector describing levels of 5 types of content consumption activity, namely read code, login, open thread, preview comment (pre-comm) and run code. The size of the state space is reduced by converting the raw

Table 1: **An excerpt from the action policy, showing its proposed content creation actions for the most commonly observed content consumption states**

read code	login	open thread	pre-comm	run code		comment	reply	share	grade-comm	grade code
0	0	0	0	0	→	0	0	0	0	1
1	0	0	0	0	→	0	0	0	0	1
2	2	2	2	2	→	2	0	0	0	0
1	0	0	1	1	→	2	0	0	0	0
2	1	2	2	2	→	0	0	2	0	0
State						Action				

log counts (e.g. number of times 'read code' happened) into 3 bands indicating low (0), medium (1) or high (2) activity relative to other time slices. For example, a state of 01012 would indicate low read code (0), medium login (1), low open thread (0), medium preview comment (1) and high run code (2). Some states are shown in Table 1. We define action as a 5 dimensional vector describing the levels of 5 content creation activities, namely comment, reply, share, grade comment (grade-comm) and grade code. As for state, they are reduced to low, medium or high relative to other time slices. For example, an action of 01012 would mean low comments, medium replies, low shares, medium comment grading, and high code grading (where we refer to the amount of grading, not the grades themselves). We can then gather observations of state-action-state tuples in the dataset, i.e. we observed this action being taken in this state and it was followed by this state. This is converted to a state transition matrix, an example entry in which is: "20020x00000" : "02002" : 0.5,"02222" : 0.5. In this example, state 20020 and action 00000 are observed to be followed by states 02002 and 02222 in an equal number of cases.

The next part of the MDP formulation is the reward function which involves assigning a value to every possible state plus a reward and cost for every possible state-action pair. State values are essentially sums of the elements of the state vector (state 00120 is worth 3). The values of state-action pairs are a sum of the values of all states observed to follow that state-action pair weighted by the number of observations of each follow on state. The cost for an action is calculated based on the frequency of that action in the observed set of actions, where we assume that infrequent actions are costly.

3. SIMULATION

Having derived an action policy, we will now evaluate it by runing it in simulation against a state transition matrix derived from a different period of the case study than the period the action policy was trained on. In this case, the training and test sets contained data derived from the same student cohort, just gathered during different time periods. The aim of this simulation is to examine the ability of the policy to generalise (to the same students in a different time window), and therefore to assess the potential usefulness of this system in a real world context.

Figure 2: **Test performance of the action policy in simulation vs. real world performance with varying time slice length. With 1 hour time slices, the MDP provides 1.25 times more value. Error bars are based on standard deviation over 100 simulation runs.**

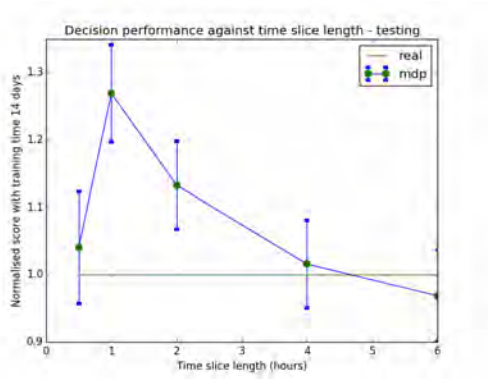


Figure 2 shows the results of running the action policy in simulation, where the training was carried out with varying time slice lengths. For each time slice length, the simulation was run 100 times to establish the typical range of performance. The one hour time slice provides the best performance, where the accumulated state values over the simulation were 1.25 times the value accumulated in the real case study data. It should be noted that the deteriorating performance as time slice length increases is likely to be caused by the smaller number of samples: there are less 6 hour slices than there are 1 hour slices. This means the transition matrix becomes very sparse, resulting in very limited simulation detail. This positive result demonstrates that the MDP approach could be a viable method to model and advise about online educational systems based around content consumption and creation.

4. CONCLUSION

We have described how social learning activity data can be formulated into an MDP and that this formulation allows the derivation of an action policy that can be used to decide what kind of content to create and when to create it, in order to maximise content consumption activity. We have also presented a preliminary validation of the action policy in a simulation based on real data, showing that the action policy selects actions that lead to higher levels of content consumption.

5. REFERENCES

[1] G. Durand, F. Laplante, and R. Kop. A Learning Design Recommendation System Based on Markov Decision Processes. In *17th ACM SIGKDD conference on knowledge discovery and data mining*, 2011.

[2] T. Snijders. Methods for longitudinal social network data: Review and Markov process models, 1995.

[3] M. Yee-King, M. Krivenski, H. Brenton, and M. D'Inverno. Designing educational social machines for effective feedback. In *8th International Conference on e-learning*, Lisbon, 2014. IADIS.

Predicting student grades from online, collaborative social learning metrics using K-NN

Matthew Yee-King
Computing, Goldsmiths
University of London
m.yee-king@gold.ac.uk

Andreu Grimalt-Reynes
Computing, Goldsmiths
University of London

Mark d'Inverno
Computing, Goldsmiths
University of London
dinverno@gold.ac.uk

ABSTRACT

We describe a collaborative video annotation system that aims to engage learners in a focused, collaborative process of content sharing and discussion, and explain how it was used in an online creative programming MOOC on Coursera. We explore the use of K-NN (K nearest neighbour) to predict which of a variable number of evenly spaced, final grade bands students will fall into based solely on a feature vector consisting of the total number of UI click and mouseover events they generated during the course. We were able to classify students into pass/fail bands with 88% precision; with 3 grade bands, precision was 77%, going down to 31% with 10 grade bands. Typically, a feature subset containing less than half of the available features provided the best performance.

Categories and Subject Descriptors

K.3.1 [**Collaborative learning**]: K.3.2 Computer science education

1. INTRODUCTION

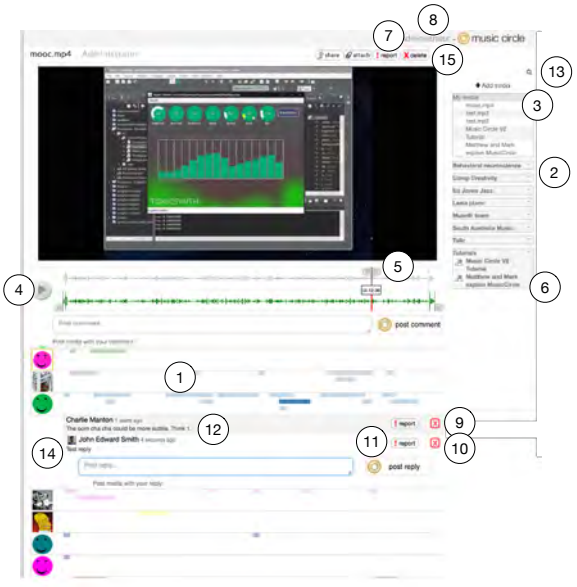
Our work is concerned with the development and analysis of systems that enable online, collaborative learning driven by different types of feedback. In this paper, we show how it is possible to predict student grades using user interface telemetry data gathered from a case study involving 993 students who completed all assessments for a creative programming course on MOOC platform Coursera. The students used a collaborative video annotation tool as part of their peer assessment, which we developed as part of an EU funded research project [5]. Previous work with collaborative media annotation systems and grade prediction includes [1, 3] and [2, 4] respectively[1].

2. THE CASE STUDY AND DATA SET

[1]A full version of the paper can be found at http://dx.doi.org/10.13140/RG.2.1.4525.9129

Three times during the course, the students were set a graded peer assessment wherein they had to extend our example programs and create a 5 minute video of themselves explaining their code and running their program. The videos were uploaded to our collaborative video annotation system wherein they could look at each others' videos and create annotations along a 'social timeline'. The system logged click and mouseover events on the UI elements shown in Figure 1, 3,716 unique users logged into our system. Of these, 3558 viewed one or more videos, 827 made one or more comments, and 258 made one or more replies to comments. 2,898 videos were submitted for three separate assessments, and were viewed a total of 112,189 times. 7,370 comments were made, and 978 replies. For this paper, we filtered the data down to all logged click and mouseover events for students who gained a final grade on the course, a total of 993 students.

Figure 1: **A screen shot of the video annotation and discussion system. The numbered labels show all of the elements of the UI for which events are logged automatically.**

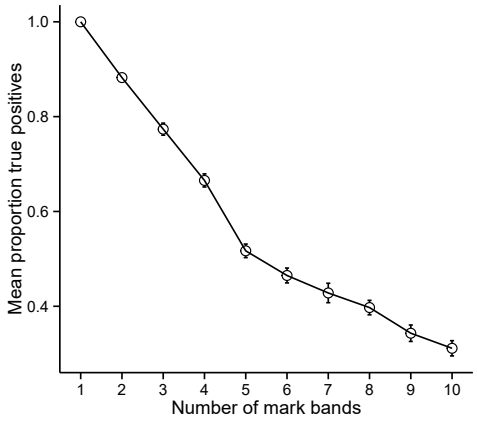


3. ANALYSIS

To predict student grades, we created a 16 dimension feature vector consisting of total numbers of clicks and mouseover events on each of the GUI elements shown in Figure 1 plus the final grade achieved by the student. We began by attempting to correlate individual elements of the feature vector to the final grade but individual correlations were too weak to predict grades, ranging between 0.53 and 0.18. This motivated us to try a multivariate classification approach. For our first analysis, we assigned labels to the students based on which of N evenly spaced grade boundaries they fell into. For example, if $N = 2$, then students were labelled **1** if $final\_grade < 50$ and **2** if $final\_grade \geq 50$. We split the dataset into equally sized training and test sets and attempted to train a K-NN classifier to assign labels to the test set, with varying numbers of mark bands and multiple run cross validation.

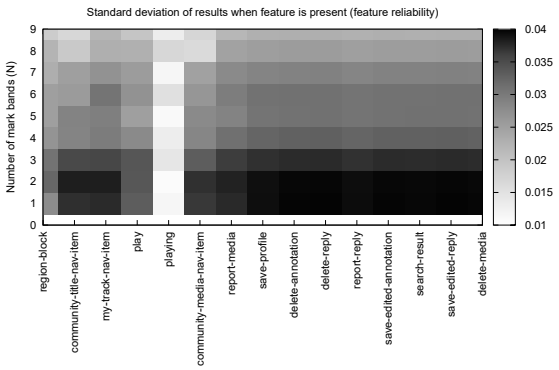
Figure 2 shows the proportion of correctly assigned labels in the test set as number of mark bands N varies from 1 to 10. For example, the pass/fail classification where $N = 2$ achieved 88% true positives. We note that the distribution of marks across the bands has a significant impact on the meaning of accuracy, and that for $N = 2$, for example, there are a large number of examples in each class which are being correctly classified.

Figure 2: **Performance of the classifier with $k = 6$ and number of mark bands $N = 1...10$.**



For our second analysis, we tried out all possible combinations of feature elements to see which combination achieved the highest classification accuracy. Since the number of features was 15, the number of permutations was 32768 (2^15). K was set to 6 and number of mark bands N varied from 1-10. Figure 3 highlights the most reliable features in the feature set by showing how much the prediction score varied (the standard deviation) across the set of all permutations per N which involved that specific feature. To be clear, it does not differentiate between features that reliably provide good or bad results. The most reliable feature was 'playing', which is triggered automatically while a video is playing. The second most reliable feature was 'region block', which is logged when a user clicks on a comment on the timeline to open the discussion thread. More work is needed to understand this result more deeply.

Figure 3: **Heat plot showing the standard deviation in the prediction results when different features are present. Low variation (lighter) is desirable, meaning a feature makes a reliable contribution to the results.**



4. CONCLUSION

We have briefly described a collaborative video annotation tool we have developed. Using interface telemetry data gathered describing click and mouseover events generated by the user interface of the system, we were able use a K-NN classifier to classify students into pass/fail bands with 88% precision; with 3 grade bands, precision was 77%, and with 10 bands it was 31%. We measured the prediction power of different combinations of the features and were able to identify the most reliable features, which relate to playing back videos, exploring content menus and reading comments.

5. REFERENCES

[1] D. Bargeron and J. Grudin. Asynchronous Collaboration Around Multimedia Applied to On-Demand Education. *Journal of Management Information Systems*, 18(4):117—-145, 2002.

[2] C. a. Coleman, D. T. Seaton, and I. Chuang. Probabilistic Use Cases: Discovering Behavioral Patterns for Predicting Certification. *Proceedings of the Second (2015) ACM Conference on Learning @ Scale - L@S '15*, pages 141–148, 2015.

[3] E. F. Risko, T. Foulsham, S. Dawson, and A. Kingstone. (CLAS): A New TOOL for Distributed Learning. 6(1):4–13, 2013.

[4] T. Sinha, P. Jermann, N. Li, and P. Dillenbourg. Your click decides your fate: Leveraging clickstream patterns in MOOC videos to infer students' information processing and attrition behavior. 2014.

[5] M. Yee-King, M. Krivenski, H. Brenton, and M. D'Inverno. Designing educational social machines for effective feedback. In *8th International Conference on e-learning*, Lisbon, 2014. IADIS.

Meta-learning for predicting the best vote aggregation method: Case study in collaborative searching of LOs

Alfredo Zapata[1], Victor H. Menéndez[1], Cristóbal Romero[2], Manuel. E. Prieto[3]
[1]Autonomous University of Yucatan, Faculty of Education, 97305, Mérida, Mexico
[2]University of Cordoba, Dept. of Computer Science, 14071, Córdoba, Spain
[3]University of Castilla-La Mancha, Computer Science Faculty, 13071, Ciudad Real, Spain
{zgonza, mdoming}@correo.uady.mx, cromero@uco.es, manuel.prieto@uclm.es

ABSTRACT
The problem of recommending learning objects to a group of users or instructors is much more difficult than the traditional problem of recommending to only one individual. To resolve this problem, this paper proposes to use meta-learning for predicting the best voting aggregation strategy in order to automatically obtain the final ratings without having to reach a consensus between all the instructors. We have carried out an experiment using data from 50 groups of instructors doing a collaborative search of LOs in AGORA repository.

Keywords
Meta-learning, Classification, LOs Collaborative Search

1. INTRODUCTION
Nowadays, there is a wide variety of e-learning repositories that provide digital resources for education in the form of Learning Objetcs (LOs). The search for and recommendation of LOs are traditionally viewed as a solitary and individual task but this is changing. On the one hand, collaborative search can be more effective than an individual search, for example in our case, a group of instructors may be interested in searching and selecting together the educational resources most appropriate to develop a new digital course. On the other hand, the goal of group recommendation is to compute a recommendation score for each item (in our case, each LO) that reflects the interests and preferences of all group members. The problem is that all group members may not always have the same tastes, and a consensus score for each item needs to be carefully designed. So, to recommend to user groups is more complicated than recommending to individuals [2]. The main problem that group recommendation needs to solve is how to adapt to the group as a whole, based on information about individual users' likes and dislikes. A solution is to use group decision strategies or aggregation methods that are inspired by social choice theory, and establish different automatic ways of how a group of people can reach a consensus. However, groups are very diverse, and no single group decision strategy works best for all groups. A way to address this issue is to identify the inherent characteristics of different groups and to determine their impacts on the group decision process [1]. Following this idea, in this paper we propose to use meta-learning for predicting the best aggregation method recommended for a group based on its characteristics. In this way, the traditional time-consuming consensus-taking among users can be avoided by using an automatic method based on meta-learning.

2. PROPOSED METHODOLOGY
In order to resolve the problem of determining which aggregation method is the most appropriate for each type of collaborative search group, we propose to use a meta-learning process (Fig. 1). The idea is to obtain automatically the aggregation method which provided/gave the best performance for a group of instructors based on its characteristics and previous rating of other similar groups. As seen in Fig. 1, the meta-learning process starts from a dataset which contains descriptive information about groups, the individual ratings of each member to all the LO's selected by the group during the collaborative search, and the consensus about the final rating assigned to all selected LO's. Next, the groups' characteristics are defined and the performance of the rating aggregation methods is evaluated in order to form a new metadata set. Then we select a classification algorithm that it used each time we have a new group of users/instructors in order to can recommend an aggregation method of their LO's rating.

Firstly, in order to create metadata, we use the following previously proposed descriptors or characteristics [1]: group size, social contact level, experience level and dissimilarity level. Additionally, we also propose a new descriptor based on the activity level of the group members in using LO repositories. Then, an evaluation phase is necessary in order to determine which aggregation method obtains the lowest error with respect to the actual consensual final rating of group members for all LOs. This actual or real rating is the final score of the group, obtained after consensus between all the members. So, it is necessary that the group have an in-person reunion or online communication in order to achieve the final score, starting with each individual rating/score and opinion. Various aggregation methods can be used to automatically obtain the final group rating for each LO [2]. We propose to use eight traditional aggregation methods

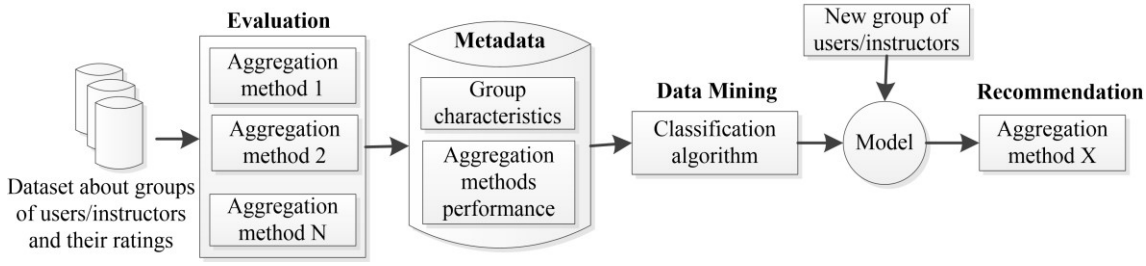


Figure 1. Meta-learning process for recommending a voting aggregation method.

(plurality voting, average, median, approval voting, least misery, most pleasure, average without misery, and fairness) plus three new weighted versions (active, social and experience user) of the average method based on [3]. In our case, instead of assuming equal weights for all the members, we give more weight to some users based on their characteristics, assuming that some members are more influential and can persuade others to agree with them. Next, a new metadata set is created by using both the characteristics of each group and the obtained aggregation method that provided the best group performance. After that, a classification algorithm is used to predict which aggregation method is most appropriate for a new group, given its characteristics. However, because there are a lot of classification techniques, we must therefore select a representative number of classification algorithms in order to compare their performance when using our metadata set. Finally, the classification algorithm that provides a better general performance will be the one selected for predicting the aggregation method most appropriate for each new group. In this way, the classification model obtained by the selected algorithm will be used for selecting, in real time, the best aggregation method for a new group according to the characteristics of the group and their individual ratings.

3. EXPERIMENTAL WORK

We have carried out an experiment in order to test our proposal of predicting the most appropriate aggregation method to use with a new group, based on the characteristics of the group members and the previous rating of similar groups. We have used data from a collaborative search of LOs in DELPHOS system [5]. We sent invitations, without using any incentive, to all instructors and final-year students of the Faculty of Education of the Autonomous University of Yucatan in Mexico to participate in the experiment. Only 75 users accepted our invitation: 27 professors or university teachers at different levels (assistant, associate and full) and 48 final-year students. We defined a total of 50 different groups of instructors and students with different typologies on their characteristics. We created a metadata set that contains both the previous characteristics/descriptors of the 50 groups as well as the best aggregation methods for each group by evaluating the performance of the 11 used rating aggregation strategies (see Table 1). In order to do this, we have used RMSE (Root-Mean-Square Error) of each aggregation method in each group. Starting from this metadata set, it is possible to predict the best aggregation method to a new group by using a classification algorithm. This is a classification in which the class or attribute to predict is precisely the aggregation method that obtains the best ranking. To this end, we have used different classification algorithms provided by the WEKA software, which is one of the most popular and most used tools for data mining. We have selected a representative number of the best known classification algorithms available in WEKA: JRip (implementation of RIPPER algorithm), J48 (implementation of C4.5 algorithm), NaiveBayesSimple (implementation of Bayes classifier), SMO (implementation of support vector classifier) and IBk (implementation of KNN or Nearest Neighbours algorithm). We have executed the previous five classification algorithms using their default parameter values and 10-fold cross-validation. In order to evaluate the classification performance and to determine the best algorithm for each group, we have used two measures that have previously been used to evaluate classification algorithm recommendation methods [4]. The first is called ARE (Average Recommendation Error) and it measures the average error of the current recommendation (predicted aggregation method) regarding the best and the worst recommendation (best and worst aggregation methods from the list of methods ordered from the lowest to the highest RMSE). The second measure is the Reciprocal Average Hit Rate, also known as Mean Reciprocal Rank (MRR), which measures the median position occupied by the method currently predicted for each of the groups in the complete list of methods ordered by RMSE.

Table 1. Average Recommendation Error and Mean Reciprocal Rank obtained by the 5 classification algorithms.

Algorithm	ARE	MRR
IBk	0,9418	0,3506
J48	0,9492	0,4239
JRIP	0,9594	0,5453
NaiveBayes	0,9458	0,4113
SMO	0,9583	0,4689

As we can see in Table 1, IBk was the best classification/prediction algorithm (followed by NaiveBayes and J48) because it obtained the lowest value of Average Recommendation Error and the lowest value of Mean Reciprocal Rank. So, since the algorithm IBk achieved the best results, it is our selected classification algorithm to automatically recommend the best aggregation method of the most similar group or nearest neighbours to every new group as the best method for rating all the LOs added to the group. In this way, the moderator of the group would use the recommended aggregation method obtained by the IBk algorithm instead of having to conduct the traditional consensual decision process.

4. ACKNOWLEDGMENTS

The authors gratefully acknowledge the financial subsidy provided by Spanish Ministry of Science and Technology TIN2014-55252-P.

5. REFERENCES

[1] Gartrell, M., Xing, X., Lv, Q., Beach, A., Han, R., Mishra, S., Seada, K., 2010. Enhancing group recommendation by incorporating social relationship interactions. In: *Proceedings of the 16th ACM GROUP '10*, ACM Press, New York, NY, USA, pp. 97–106.

[2] Masthoff, J., 2011. Group recommender systems: combining individual models, in: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), *Recommender Systems Handbook*. Springer Press, New York, pp. 677–702.

[3] Popescu, G., 2013. Group recommender systems as a voting problem. In: *Proceedings of the 5th International Conference on Online Communities and Social Computing*, OCSC 2013, Springer, Berlin, pp. 412–421.

[4] Song, Q., Wang, G., Wang, C., 2012. Automatic recommendation of classification algorithms based on data set characteristics. *Pattern Recogn*. 45(7), 2672–2689.

[5] Zapata, A., Menéndez, V.H., Prieto, M.E., Romero, C., 2013. A framework for recommendation in learning object repositories: an example of application in civil engineering. Adv. Eng. Softw. 56, 1–14.

Soft Clustering of Physics Misconceptions Using a Mixed Membership Model

Guoguo Zheng
University of Georgia
Athens, GA
ggzheng@uga.edu

Seohyun Kim
University of Georgia
Athens, GA
seohyun@uga.edu

Yanyan Tan
University of Georgia
Athens, GA
yanyan.tan25@uga.edu

April Galyardt
University of Georgia
Athens, GA
galyardt@uga.edu

ABSTRACT

Students often possess multiple, conflicting misconceptions which may be activated and expressed in different contexts. In this paper, we use a mixed membership model to explore the patterns of misconceptions in introductory physics. Mixed membership models have been widely used for modeling observations that have partial membership in several latent groups. The latent groups in the current study are misconception patterns. This model allows us to examine whether students are likely to hold a few or many misconceptions, as well as which misconceptions are likely to co-exist. Physics knowledge was measured with the Force concepts inventory (FCI). We found three dominant response patterns, with different misconceptions prominent within each pattern.

1. INTRODUCTION

Student misconceptions can be persistent, and interfere with learning unless they are addressed directly. One important characteristic of misconceptions is that students possess many different knowledge components simultaneously, so that the particular schema or rule a student uses to solve a question depends on many different factors, including the context of the question [4]. This paper presents a case-study for using a mixed-membership model [1] to capture the characteristics and coherent patterns among students' misconceptions in introductory physics. Mixed membership model allows students to possess different misconception patterns (profile) across test questions. In this study, we focus on two questions: (1) What are the common misconception pattern students possess across the test, and which misconceptions tend to co-occur. (2) How much does each student exhibit each pattern?

2. METHODS

2.1 Mixed membership model

Mixed membership models allow an individual to switch profiles across contexts, test items. How much each individual uses each profile is parametrized by $\theta_i = (\theta_{i1}, \ldots, \theta_{iK})$. The components of θ_i are nonnegative and sum up to 1. Z_{ij} indicates the profile that student i uses for item j, so that

$$Z_{ij}|\theta_i \sim Multinomial(\theta_i).$$

Each latent profile has its own probability distribution for observed variables. Since the items from the case study are multiple choice, if X_{ij} denotes the observed response for student i on item j, then $X_{ij}|Z_{ij} = k \sim Multinomial(\beta_{(j|Z_{ij}=k)})$, where $\beta_{(j|Z_{ij}=k)} = (\beta_{kj1}, \ldots, \beta_{kjm}, \ldots, \beta_{kjM})$, β_{kjm} denotes the probability that a student using profile k on item j will select option m, and M is the number of options.

In the mixed membership model, the generative process is given by [5,6]:

1. For each item $j = 1, \ldots, J$, draw $\beta_{(j|Z=k)} \sim Dirichlet(\eta)$, for $k = 1, \ldots, K$.
2. For each individual $i = 1, \ldots, N$
   (a) Draw $\theta_i \sim Dirichlet(\alpha)$
   (b) For each item $j = 1, \ldots, J$,
      i. Draw $Z_{ij}|\theta_i \sim Multinomial(\theta_i)$.
      ii. Draw $X_{ij}|Z_{ij} \sim Multinomial(\beta_{(j|Z_{ij}=k)})$,

Here η and α are prior parameters. These could be estimated in an empirical-Bayes fashion. We choose to set these parameters to incorporate prior information, and stabilize the model.

2.2 FCI Data

From 1995-1999, 4450 high school students responded to The Force Concept Inventory (FCI), one of the most commonly used assessments in physics to measure students' understanding of concepts on Newtonian mechanics. We foucsed on the pre-test scores from a larger study [3]. The FCI consists of 30 multiple-choice items, with 18 items measuring *Newton's Second Law*. Most of the distractor options on this test were designed to map to a common physics misconception, though some distractors are statements that cannot be

explained by physics theories. More detailed explanation of these misconceptions can be found in [2].

3. RESULTS

We estimated the mixed membership model using MCMC with 5,000 iterations (1,000 burn-in). We placed a weakly informative prior on $\beta_{(j|Z=1)}$, of $\eta_{j1} = (50, 1, 1, 1, 1)$, and a flat prior to all the other parameters.

3.1 Number of Profiles

We fit mixed membership model with three to seven profiles. The same misconceptions were found to co-exist regardless of the number of profiles. In the 3-profile model, students have the most distinct probabilities of selecting a particular response across profiles, and were more likely to exclusively belong to one of the profiles ($\theta_{ik} > 0.8$). Thus, we can say that three profiles is representative of students' misconception patterns and in this paper, we focus on the 3-profile model.

3.2 Students' Membership in the Profiles

Profile memberhsip of each student is captured by the parameter $\theta_i = (\theta_{i1}, \theta_{i2}, \theta_{i3})$ shown in Figure 1. The proportion of students who exclusively belong to profile 3 is the highest, followed by profile 2 and profile 1. There are many students who are between profile 2 and profile 3 as well as between profile 3 and profile 1. Far fewer students fall between profile 2 and profile 1.

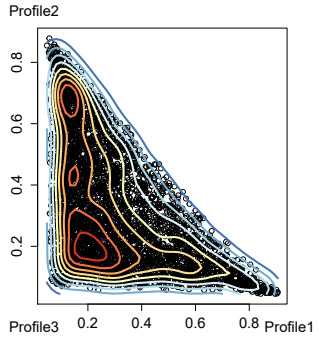


Figure 1: Contour map of posterior distribution for students' membership in the three profiles. X nad Y axes represent θ_{i1} for Profile 1 and Profile 2 (θ_{i1}) respectively. Profile 3 can be obtained by $\theta_{i3} = 1 = \theta_{1i} - \theta_{i2}$

3.3 Characteristics of Profiles

Each profile is parameterized by a probability distribution over the responses to each item, $\beta_{(j|Z=k)} = (\beta_{kj1}, \ldots, \beta_{kj5})$. We illustrate the characteristics of each profile using items that measure Newton's Second Law of Motion, and these characteristics hold up for all the items in the FCI instrument.

Misconception Profile (profile 3) This profile is characterized by high probability on responses containing misconceptions. Recall also, that this profile had the most students that belonged to it exclusively, as well as large numbers of students who were between it and the other profiles (Figure 1). In this profile, some misconceptions, such as *impetus dissipation* are observed repeatedly across items. However, we also observe that the activation of a misconception depends on items. For example, the misconception *impetus supplied by "hit"* is likely to be observed in item 30 even though it is also associated with item 11. This profile has the most profound implications for instruction since it is the largest, and demonstrates that students tend to not hold a single misconception, but rather many misconceptions that co-exist and may be expressed in different contexts.

Mostly Correct Profile (profile 1). This profile places a high probability on the correct response for most items, and has the smallest number of students that have high membership in the profile. However, on a few items, this profile is also associated with misconceptions. Some of these misconceptions, such as *largest force determines motion* were shared by the other profiles which instructors will want to address, and some of them tend to be of a higher-level.

Uniform Profile (profile 2). In general, the probability of choosing an option was similar across at least three options for most of the items. This profile has a large number of students who belong almost exclusively to it. Even when we increased the number of profiles, it did not disappear, nor decompose into separate profiles. These observations indicate that students in this profile do not have any coherent pattern in their responses.

4. CONCLUSION AND DISCUSSION

This study illustrates how mixed membership models can be a good tool to summarize a number of misconceptions into fewer numbers of profiles by identifying misconceptions that are likely to co-exist. Among the three profiles we found with FCI data, the majority of students had partial or complete membership in the *misconception profile*. The high coherence of co-existing misconceptions across a large number of students in this profile demonstrates the real power of this mixed membership analysis. By finding coherent patterns exhibited by many students at least some of the time, we find evidence that may suggest new theory. Future work can focus on the challenge of deciding an optimal number of profiles when conducting mixed membership models and the assumption that Z_{ij} depends on both i and j. Profile transitions between pre- and post-test should also be examined.

5. REFERENCES

[1] E. Erosheva, S. Fienberg, and J. Lafferty. Mixed-membership models of scientific publications. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5220–5227, 2004.

[2] D. Hestenes and J. Jackson.

[3] D. Hestenes, M. Wells, G. Swackhamer, et al. Force concept inventory. *The physics teacher*, 30(3):141–158, 1992.

[4] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798, 2012.

Perfect Scores Indicate Good Students !? The Case of One Hundred Percenters in a Math Learning System

Zhilin Zheng
Department of Computer Science
Humboldt-Universität zu Berlin
Berlin, Germany
zhilin.zheng@hu-berlin.de

Martin Stapel
Department of Computer Science
Humboldt-Universität zu Berlin
Berlin, Germany
martin.stapel@hu-berlin.de

Niels Pinkwart
Department of Computer Science
Humboldt-Universität zu Berlin
Berlin, Germany
niels.pinkwart@hu-berlin.de

ABSTRACT
As a teacher or administrator, seeing a student scoring 100% in an exercise series within an online learning system would typically raise no immediate worries. This paper analyzes the "one hundred percenter" sessions in a math learning system. We argue that some student sessions with 100% score may actually not be predictive of student's learning success, and that a frequently exhibited student strategy of getting a perfect score by skipping exercises and repeating series is not ideal.

Keywords
Learning Analytics; Educational Data Mining; User Modelling; Student Behavior; Gamification

1. INTRODUCTION
Many educational technology systems allow students to take exercises multiple times and thus follow a resubmission policy [4; 6]. In this model, students have a chance to revise their answers by looking closely at their errors and the system gives feedback accordingly (which may vary in form and degree of detail). This resubmission policy certainly benefits self-regulated learning. Some of these learning systems limit the number of resubmissions, whereas others leave it unlimited [6]. Nevertheless, a possible negative side effect of this policy is evident as well. Under a resubmission policy, students can potentially take a trail-and-error strategy with little or even no thinking about the exercises and still try to get a high score [1; 4]. To address this issue, randomized initial data can be used to generate new (but structurally similar) exercises and thus avoiding repetitive occurrences of same exercises [5]. This strategy has shown to have a positive impact on students' learning results [6].

In this paper, we conduct an investigation in the context of a math learning system with a feature of resubmission. Log files indicate that a portion of students were eager to achieve a 100% success rate by taking a strategy of skipping exercises with a 'help' of resubmission. As far as we know, this phenomenon has not been studied extensively up to now. Nevertheless, skipping behavior itself is quite common in computer-supported learning systems. If a resubmission policy is allowed, restarting an exercise series or a quiz is technically possible and not as expensive as in paper-and-pencil tests in physical classroom settings. One may argue that students' motivation of achieving a 100% success is not surprising too. In a traditional classroom this happens quite often because students desire their teacher's praise or want to show off their talent with such a high learning performance. In this paper we thus do not primarily intend to discuss the phenomenon as such, but want to investigate two related questions. First, is this skipping strategy (aborting and restarting an exercise series after a mistake) actually a fast way to achieve a 100% success score, or are there more efficient strategies to reach this goal? Second, from a pedagogical viewpoint, do students who take this strategy perform as good as their learning outcomes seem to indicate – i.e., perfectly?

2. DATA
Bettermarks[1] is an online math learning system. It delivers math learning content in cooperation with K-12 schools (grades 4-10). Since the system provides flexibility to choose math topics and exercise series according to needs of different curriculums, it is frequently blended into classroom teaching by school teachers. Typically, teachers assign exercises (organized in exercise series) to their students and their achievement is in turn reported back to the teachers via the system. Bettermarks employs an unlimited resubmission strategy, which means that students can make as many attempts as they want. With such a feature, students are expected to iteratively make use of more attempts to correct their errors with helps of the system's feedback and/or hints.

After a close look at the sever log file, we found that plenty of the students made many skipping attempts before a 100% success. We termed such an interesting phenomenon as a "one hundred percenter with skipping". They did not take the exercises one after another as some of their peers did. Instead they skipped all the remaining exercises and made a new attempt once an error occurred. From January 2014 till November 2014 we found 8,640 (6.4%) sessions involved in such a phenomenon out of totally 687,688 sessions.

3. ANALYSES AND RESULTS
We identified another two different groups of student sessions with least one 100% success in one attempt of the exercise series. One group is the sessions without any skipping behavior but at least a 100% success once (59,941 in total). The other group contains sessions with a 100% success at the first attempt, but still with next attempts in the same exercise series. We termed this group "strong one hundred percenters" (3,854). The one hundred percenters with skipping showed a totally different learning style than their counterparts without skipping. Upon realizing a problem (e.g., a mistake made or an apparent difficult

[1] http://bettermarks.com/

exercise), the former group decided to skip over this exercise and the remaining ones in the series, and restarted the series. To the contrary, the ones without skipping chose to continue with the current work. They took every learning chance (as the system designer or the teacher would probably have hoped). Through this behavior, they could still probably learn something from the feedback or the next exercises in the series even though they had made an error. However, their desire to achieve a 100% success was evident through their behavior. The question which style (with or without skips) leads to the shared goal (100% success) quicker is interesting. To answer this question, we counted the students' attempts to a 100% success respectively. Students with the skipping strategy in fact needed more attempts to achieve their desired perfect score (3.6 attempts vs 2.4 attempts). This difference is statistically significant (Welch's t-test with different variance, $p<0.001$). In other words, students that chose to do all the exercises instead of skipping achieved a 100% success faster. Note that we took the number of attempts as a measure instead of time spent because that would bring individual's faster or slower learning pace as a noise into our analysis.

Interestingly, some of the one hundred percenters continued with their learning activities even after having obtained a perfect score. They even made more attempts right after their achievement of 100% success. In this case, we can hypothesize that the reward-oriented motivation was lower than the intrinsic, learning-oriented motivation: the system would reward students achievement badges once they achieved a 100% success but no more afterwards. We got 129 (1.4%) of such sessions out of the one hundred percenters with skipping, 1,414 (2.3%) sessions out of the one hundred percenters without skipping, and 3,854 (by definition, 100%) sessions out of the strong one hundred percenters. Solely from the participation we can intuitively see that very few one hundred percenters with skipping engaged in their learning activities once they had got the achievement badges in comparison of another two groups. We sought to investigate their learning performance under this situation (only with intrinsic motivation). We calculated their average success rate over attempts after that 100% success attempt. The average learning performance of one hundred percenters with skipping (0.78) is much lower than without skipping (0.91). Unsurprisingly, the strong one hundred percenters take the leading position (0.94). A Kruskal-Wallis H-test confirmed significant difference ($p<0.001$).

We can now give some answers to our questions stated in Section 1. First, the skipping strategy does not show any advantage when compared to the non-skipping strategy. To the contrary, students who take this strategy needed more attempts to achieve a 100% success at the end. More importantly, one hundred percenters with skipping reveal significantly weaker capabilities than their peers during the attempts after a 100% success. This would put this portion of students at risk especially when teachers only take their best outcome as a rating criterion. Since they do not show any weakness solely on that indicator, their teachers would overlook them (assuming they do fine) and move their attention to the weak students. As such, one hundred percenter behavior with skipping is not a fruitful strategy – it does not make the process of getting the 100% badge more efficient, and in fact students that pursue this strategy did not learn as much as their scores indicate, and less than their peers.

4. CONCLUSION

This work analyzes a portion of students in a math learning environment who achieve a 100% success in an exercise series through skipping exercises and then repeating the series. A closer look at the data in the learning system yielded several insights. The first one is that the adoption of the skipping strategy does not help to speed up to a 100% success. Instead, a non-skipping strategy leads students to achieve a perfect score faster. Another yet more important finding is that one hundred percenter behavior could put students at risk of being overlooked by teachers. They actually do not perform as excellent as their learning performance indicates.

With regard to the motivation of one hundred percenters, achievement badges available in the system, a gamification strategy often used in educational systems, could explain their motivation. Still there could be some other incentives, for example, encouragement or rewards coming from somewhere outside of the learning system. The learning system we studied is integrated into blended teaching settings in most cases. Thus teachers should have much space to motivate their students without a need to solely rely on the learning system's rewarding strategy. Apart from motivation factors, carelessness or a slip [2; 3] could explain one hundred percenters' skipping behavior as well.

5. ACKNOWLEDGMENTS

Our thanks to the Chinese Scholarship Council (CSC) for funding the first author's research.

6. REFERENCES

[1] AUVINEN, T., 2015. Harmful Study Habits in Online Learning Environments with Automatic Assessment. In *Learning and Teaching in Computing and Engineering (LaTiCE), 2015 International Conference on*, 50-57.

[2] BAKER, R.S., CORBETT, A.T., and ALEVEN, V., 2008. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *Intelligent Tutoring Systems* Springer, 406-415.

[3] HERSHKOVITZ, A., HERSHKOVITZ, R.S.J., DE BAKER, J., GOBERT, M., WIXON, M.S., and PEDRO, 2013. Discovery With Models: A Case Study on Carelessness in Computer-Based Science Inquiry. *American Behavioral Scientist 57*, 10, 1480-1499.

[4] KARAVIRTA, V., KORHONEN, A., and MALMI, L., 2006. On the use of resubmissions in automatic assessment systems. *Computer Science Education 16*, 3 (2006/09/01), 229-240.

[5] KORHONEN, A. and MALMI, L., 2000. Algorithm simulation with automatic assessment. In *Proceedings of the Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSEconference on Innovation and technology in computer science education* (Helsinki, Finland2000), ACM, 343157, 160-163.

[6] MALMI, L. and KORHONEN, A., 2004. Automatic feedback and resubmissions as learning aid. In *Advanced Learning Technologies, 2004. Proceedings. IEEE International Conference on*, 186-190.

Doctoral Consortium

Towards the Understanding of Gestures and Vocalization Coordination in Teaching Context

Roghayeh Barmaki
Department of Computer Science
University of Central Florida
barmaki@cs.ucf.edu

Charles E. Hughes
Department of Computer Science
University of Central Florida
ceh@cs.ucf.edu

ABSTRACT

Nonverbal behaviors such as facial expressions, eye contact, gestures, postures and their coordination with voice tone and prosody have strong impact on the process of communicative interactions. Successful employment of nonverbal behaviors plays an important role in interpersonal communication in the classroom between students and the teacher. Student teachers need to improve their teaching skills, from communication to management, and prior to entering the classroom. To support these aspects of teacher preparation, we developed a virtual classroom environment, TeachLivE^{TM} for teacher training, reflection and assessment purposes. In this work we investigate the connections between gestures and vocalization characteristics of participants in a teaching context for two settings within the TeachLivE environment.

We have developed an immediate feedback application that is presented to the participants in one of the study settings. It provides visual cues to the participant in front of the tracking sensor any time that she exhibits a closed stance. Identification of these type of connections between acoustic and gestural components of communication provides an added dimension that could assist us in using machine learning methodologies to extract multimodal features as teaching competency measures.

Keywords

gesture; vocalization; nonverbal behavior; Microsoft Kinect; virtual teaching rehearsal environment.

1. INTRODUCTION

Interpersonal communication involves a variety of modes and components in communication. We might think that actual words are the primary part of communication; however, the majority of interaction between individuals, including students and teachers, is nonverbal, encompassing between 65 and 93 percent of what occurs related to learning [7]. These nonverbal elements include both nonvocal (e.g. body language) and vocal components (e.g. voice pitch and intonation). Body language by itself include several aspects: facial expressions, eye contact, posture or stance, gestures, touch and appearance. This research investigates the connection of postures and/or gestures with acoustic components of the nonverbal communication in the teaching context.

Multimodal analysis co-processes two or more parallel input streams (modes) from human-centered interactions that contain rich high-level semantic information [9]. Teaching and learning have always been multimodal as both are unified with speech, gesture, writing, image and spatial setting [12]. Multimodal data analysis in a teaching context helps us to have an informed understanding of the performances of the teacher participants.

TeachLivE is a simulated classroom setting used to prepare teachers for the challenges of working in K-12 classrooms. Its primary use is to provide teachers the opportunity to rehearse their classroom management, pedagogical and content delivery skills in an environment that neither harms real children, nor causes the teacher to be seen as weak or insecure by an actual classroom full of students. TeachLivE uses its underlying multi-client-server architecture called AMITIES- Avatar Mediated Interactive Training and Individualized Experience System [8]. A human-in-the loop (called an interactor) orchestrates the behavior of the virtual students in real-time based on each character's personality and backstory, a teaching plan, various genres of behaviors and the participant's input. The virtual classroom is displayed on a large TV screen to the participant and the view of the virtual classroom scene dynamically changes based on the participant's movements in front of the tracking sensor. We have developed a real-time gesture recognition application for nonverbal communication skill training, based on the Microsoft Kinect SDK [1] as part of ReflectLivE, the TeachLivE integrated reflection tool [3]. The hypothesis is that our developed feedback application has positive impact on the participants' body language, leading to more open and fewer closed stances. The open stance has arms and legs not crossed in any way. To explore the validity of this hypothesis and system usability evaluation, we report the results from the conducted case study with two settings using the feedback application (section 2.1).

We are also interested in looking at the connections between the participant's gestures and acoustic characteristics in different situations in the classroom, such as while asking questions from virtual students, conversation turn-taking after students' responses, introducing a new topic, etc. The analysis of the recorded sessions from a gesture-voice aspect is another motivation for this research that seeks a broader understanding of communication practices that reflect and support teaching competency.

Investigating the related research, there have been a number of prior attempts to develop social skill training and

feedback applications using interactive environments. Presentation Trainer [10] collects multimodal data using the Microsoft Kinect and provides immediate cues about the trainee's body posture, embodiment and voice volume during her presentation. Similarly, Dermody and Sutherland [5] present a multimodal prototype for public speaking purposes that uses the Kinect sensor. Their system provides real-time feedback on gaze direction, body pose and gesture, vocal tonality, vocal dysfluencies and speaking rate.

At first glance, gesture and speech may be coupled less directly than, e.g., prosody and speech, as both originate in very different physiological systems. However, some views and findings suggest a close connection between both, especially in production. This mutual co-occurence of speech and gesture reflects a deep association between the two modes that transcends the intentions of the speaker to communicate [11].

2. APPROACH

We present our research to understand the gesture and vocalization connections in the following two separate subsections since most of our currently reported research has been done independently with our effort to fuse the collected multimodal data still under development.

2.1 Gesture

This research evolved based on the existing literature expressing the importance of open body gesturing in successful interactive teaching (teaching competency) [2]. Reviewing the existing recordings of teaching sessions in TeachLivE gave us a baseline about the way teachers use their body in the virtual classroom. In our observations, most of the teachers were not thoughtful of their body movements and many of them exhibited closed stances most of the time in their teaching sessions. The recognized frequent closed postures (or closed gestures) were hands folded in front and back, hands on hips, and crossed arms. These gestures are noted as closed or "not-recommended" gestures. We are interested in detecting these closed gestures and reminding the trainees about their closed body language. In social skill training, the impact of immediate and real-time feedback in the rehearsal process has been reported as very positive in comparison to other types of feedback provision such as delayed feedback [10]. The developed feedback application is capable of providing visual or haptic (vibration wrist band) prompts in real-time for targeted closed gestures. The effectiveness of the implemented visual feedback application was evaluated by conducting a user study. It was a single-time within-subjects, counterbalanced study with two settings (TeachLivE with and without feedback application) and each session was 7-minute long. Participants (N=30, 6M, 24F) were asked to attend both of the settings, and complete pre and post questionnaires (the total recruitment time was approximately 45 minutes per participant). We randomly assigned the participants into two groups A and B, where group A (N=15, 3M, 12 F) experienced TeachLivE with feedback setting in their second session and group B had this experience in their first session. The collected full-body tracking data from the participants was processed [3] to extract the percentage of time that a subject exhibited closed gestures (CGP) in the recorded sessions. Our expectation based on the hypothesis (section 1) was that we would observe a considerable difference between groups A and B in the first session and a slight difference between the two groups in the closed body gesture employment in the second session. To evaluate the impact of our proposed feedback application on body language thoughtfulness, we calculated CGP for 60 recorded clips from 30 participants. The boxplot in Figure 1 presents the distribution of CGP between two groups of participants.

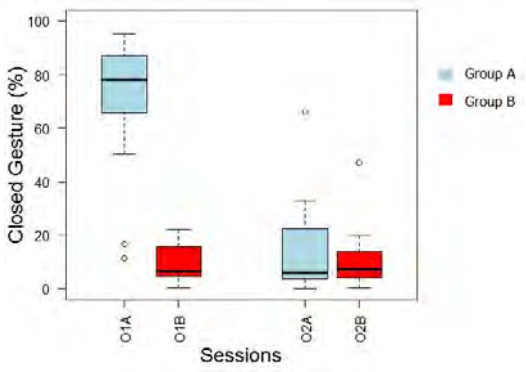


Figure 1: Medians and interquartile ranges of CGP exhibition in two sessions (observations) among groups A and B. Circle represent outliers.

Figure 1 shows some of key findings from this study. It presents the wide range (from 95% to 16%) of closed gesture employment for group A in the first session. It also indicates the median of CGP for group B participants is lower than group A (6.4 % and 7.2% for two sessions for group B and 78% and 5.9% for group A). As Figure 1 indicates, the hypothesized statement is supported for the participants of the study. The average time that all of the participants in group A exhibited closed gestures reduced significantly from their first session to their second session. Most interestingly, the participants in group B exhibited open gestures most of the time even in the second unaided session.

2.2 Vocalization

In this study, we recorded video, audio, full body tracking data and event logging information (including virtual students' talk-time and behaviors) from the TeachLivE system. The reader can find further recording details in [3].

After collecting the data, we processed the recorded audio from video sessions using Audacity software to extract the Waveform Audio File Format from recorded avi files. We opened the .wav files in the Praat tool [4] and extracted some basic vocal characteristics (pitch and intensity objects) from the audio files. Praat is a free computer software package for the analysis of speech. Voice pitch is the perceptual correlate of vocal fundamental frequency and voice intensity indicates voice loudness in db. A PitchTier object represents a time-stamped pitch contour (hereby feature), i.e. it contains a number of (time, pitch (Hz)) points, without voiced/unvoiced information. An IntensityTier object represents a time-stamped intensity contour, i.e., it contains a series of (time, intensity) points [4]. Pitch and intensity tier associated with our recorded sessions were exported for multimodal analysis purpose to the ANVIL [6]. ANVIL is a video annotation tool that offers multi-layered annotation

based on a user-defined coding scheme. Figure 2 shows the ANVIL tool.

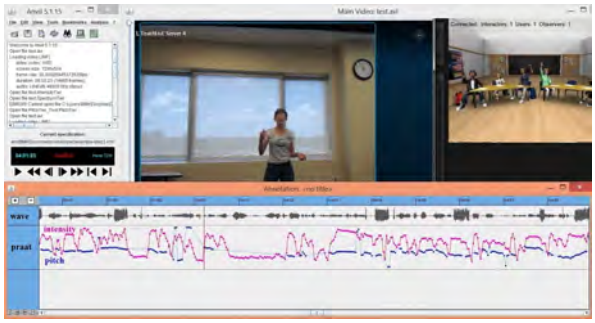


Figure 2: TeachLivE video sessions (including the participant front view and virtual classroom scene) within the ANVIL annotation tool [6]. Three acoustic contours waveform, pitch (blue) and intensity (pink) [4] are imported to the annotation project.

We intend to add our gesture recognition application output as an extended contour in the ANVIL. This will automatically present the types and timing for different closed gestures during the recorded session. The current version of the ANVIL does not support the exported (closed) labels of frames from the Kinect V2 gesture recognition tool as a contour, so we are working on this open-source tool to develop our desired contour structure. As mentioned earlier, our goal of using ANVIL is to understand the correlations of acoustic features with gesturing in these three main cases: 1) when the participant teacher asks a question from virtual classroom, 2) when the teacher listens to the responses from the class (conversation turn taking between students and teacher), and finally 3) when the teacher introduces a new or abstract topic or is summarizing the discussion. Literature supports that teachers gesture more in the mentioned cases [2]. We will annotate the recorded videos based on the teaching plan, conversational cases, open/closed, and affirmative gesture employment. The automatically generated vocalization information would be exported in conjunction with manual annotation data for further analysis.

3. CLOSING REMARKS

The study reported here fills a gap in multimodal research for education. In this paper, we first explained the impact of nonverbal behaviors in teaching competency. We then reported a case study to evaluate the performance of our developed feedback application for nonverbal communication skill training. We used the Microsoft Kinect sensor and its full-body tracking data stream to develop our real-time gesture feedback application. The results from the recorded body tracking data indicated the positive impact of informed body language and gesture in communication proficiency. We also introduced relevant tools and techniques for multimodal feature extraction for teaching competency, and we expect to report the results after developing an appropriate coding scheme framework and the annotation procedure.

For future research, we are looking forward to uncovering additional teaching evaluation insights with the analysis and evaluation of multimodal recorded data, as multimodality is an integral part of teaching.

Acknowledgments

The authors acknowledge the support of the Bill & Melinda Gates Foundation (OPP1053202) and the National Science Foundation (CNS1051067, IIS1116615). We also wish to express our gratitude to the entire TeachLivE team, especially the interactors who give life and authenticity to our avatars. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

4. REFERENCES

[1] Visual gesture builder: A data-driven solution to gesture detection. http://aka.ms/k4wv2vgb, July 2014. Retrieved 3/10/2016.

[2] M. W. Alibali and M. J. Nathan. Teachers' gestures as a means of scaffolding students' understanding: Evidence from an early algebra lesson. *Video research in the learning sciences*, pages 349–365, 2007.

[3] R. Barmaki and C. E. Hughes. Providing real-time feedback for student teachers in a virtual rehearsal environment. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 531–537, New York, NY, USA, 2015. ACM.

[4] P. Boersma and D. Weenink. Praat: doing phonetics by computer [computer program] version. 6.0.17, 2016. Accessed 5/07/2016 from http://www.praat.org/.

[5] F. Dermody and A. Sutherland. A multimodal system for public speaking with real time feedback. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 369–370, New York, NY, USA, 2015. ACM.

[6] M. Kipp. Anvil: The video annotation research tool. In *The Oxford Handbook of Corpus Phonology*. Oxford University Press, 2014.

[7] A. Mehrabian. *Silent Messages: Implicit Communication of Emotions and Attitudes*. Wadsworth, 1972.

[8] A. Nagendran, R. Pillat, A. Kavanaugh, G. Welch, and C. Hughes. A unified framework for individualized avatar-based interactions. *Presence: Teleoper. Virtual Environ.*, 23(2):109–132, Aug. 2014.

[9] S. Oviatt and P. R. Cohen. *The Paradigm Shift to Multimodality in Contemporary Computer Interfaces*. Morgan & Claypool Publishers, 2015.

[10] J. Schneider, D. Börner, P. van Rosmalen, and M. Specht. Presentation trainer, your public speaking multimodal coach. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 539–546, New York, NY, USA, 2015. ACM.

[11] P. Wagner, Z. Malisz, and S. Kopp. Gesture and speech in interaction: An overview. *Speech Communication*, 57:209–232, 2014.

[12] M. Worsley, K. Chiluiza, J. F. Grafsgaard, and X. Ochoa. 2015 multimodal learning and analytics grand challenge. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 525–529, New York, NY, USA, 2015. ACM.

Towards Modeling Chunks in a Knowledge Tracing Framework for Students' Deep Learning

Yun Huang
Intelligent Systems Program
University of Pittsburgh
210 S. Bouquet Street
Pittsburgh, PA, USA
yuh43@pitt.edu

Peter Brusilovsky
School of Information Sciences
University of Pittsburgh
135 N. Bellefield Ave.
Pittsburgh, PA, USA
peterb@pitt.edu

ABSTRACT

Traditional Knowledge Tracing, which traces students' knowledge of each decomposed individual skill, has been a popular student model for adaptive tutoring. Unfortunately, such a model fails to model complex skill practices where simple decompositions cannot capture potential additional skills that underlie the context as a whole constituting an interconnected *chunk*. In this work, we propose a data-driven approach to extract and model potential *chunk units* in a Knowledge Tracing framework for tracing deeper knowledge, which is primarily based on Bayesian network techniques. We argue that traditional prediction metrics are unable to provide a "deep" evaluation for such student models, and propose novel data-driven evaluations combined with classroom studies in order to examine our proposed student model's real-world impact on students' learning.

Keywords

complex skill, chunk, robust learning, deep learning, Knowledge Tracing, Bayesian network, regression

1. INTRODUCTION

Knowledge Tracing (KT) [4] has established itself as an efficient approach to model student skill acquisition in intelligent tutoring systems. The essence of this approach is to decompose domain knowledge into elementary skills, map each step's performance into the knowledge level of each single skill and maintain a dynamic knowledge estimation for each skill. However, KT assumes skill independence in problems that involve multiple skills, and it is not always clear how to decompose the overall domain knowledge. Recent research demonstrated that the knowledge about a set of skills can be greater than the "sum" of the knowledge of individual skills [8], some skills must be integrated (or connected) with other skills to produce behavior [11]. For example, students were found to be significantly worse at translating two-step algebra story problems into expressions (e.g., 800-40x) than they were at translating two closely matched one-step problems (with answers 800-y and 40x) [8]. Also, recent research that has applied a difficulty factor assessment [1] demonstrated that some factors underlying the context combined with original skills can cause extra difficulty, and should be included in the skill model representation. Meanwhile, research on computer science education has long argued that knowledge of a programming language cannot be reduced to simply the "sum" of knowledge about different constructs, since there are many stable patterns (schemas, or plans) that have to be taught or practiced [16]. We summarize the above findings and connect them with a long-established concept in cognitive psychology called *chunks*. According to Tulving and Craik [17], a chunk is defined as "a familiar collection of more elementary units that have been inter-associated and stored in memory repeatedly and act as a coherent, integrated group when retrieved". It has been used to define expertise in many domains since Chase and Simon's early research in chess [2]. We argue that modeling chunks is important but it hasn't been well-addressed in the current Knowledge Tracing framework. In order to identify chunks in a modern data-driven manner, we propose starting from automatic extraction of stable combinations between individual skills, or between skills and difficulty factors from huge volumes of data available from digital learning systems. We think that such *chunk units* contain different complexity levels, and more complex chunk units can be constructed from simpler chunk units, so they could and should be arranged hierarchically. So we propose a hierarchical Bayesian network which we consider a natural fit for the skill and student model, rather than alternative frameworks [1, 14, 12].

Meanwhile, complex skill knowledge modeling has been a challenge. Starting from simple variants based on traditional KT [5], more advanced models have been put forward. However, these student models use a "flat" knowledge structure, and research works that consider relationships among skills mostly focus on prerequisite relations [3] or granularity hierarchy [13]. Regarding the data-driven evaluations of student models, problem-solving performance prediction metrics [7, 5] have raised some growing concerns [6, 9]. A recent learner outcome-effort paradigm and a multifaceted evaluation framework [6, 9] offer promising methods that we plan to extend. We also plan to conduct classroom studies that deploy a new adaptive learning system that is based on our proposed student model.

2. PROPOSED CONTRIBUTIONS

The first contribution we expect to achieve is to present a novel perspective and data-driven approach for building (skill and) student models with *chunks*. Second, we aim to present a novel multifaceted data-driven evaluation framework for student models that considers practically important aspects. Third, we aim to demonstrate our proposed model's impact for real-world student learning such as helping differentiating shallow and deep learning, enabling better remediation, and ultimately promoting deep learning.

3. APPROACH AND EVALUATION

3.1 Model Construction

Our proposed student model will conduct performance predictions, dynamic knowledge estimations, and mastery decisions when deployed in a tutoring system. To save space, we only describe the major components here.

3.1.1 Representing Chunk Units

To start, we plan to use the Bayesian network (BN) framework for the final skill and student model. We call our proposed model *conjunctive knowledge modeling with hierarchical chunk units (CKM-HC)* (Figure 1).

- **The first layer** consists of basic individual skills (e.g., K_1) that capture a student's basic knowledge of a skill.
- **The intermediate layers** consist of *chunk units* (e.g., $K_{1,2}$), which can be derived from smaller units that capture deeper knowledge.
- **The last layer** consists of *Mastery* nodes (e.g., M_1) for each individual skill, which reflects the idea of granting a skill's mastery based on the knowledge levels of relevant chunk units. We now assert mastery of a skill by computing the joint probability of the required chunk units being known.

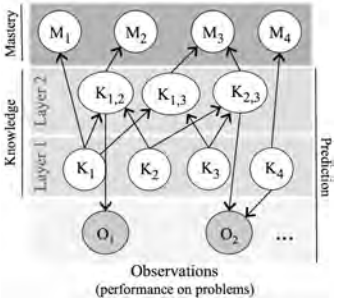


Figure 1: The BN structure of CKM-HC, with pairwise skill combinations as chunk units, in one practice time slice.

3.1.2 Identifying Chunk Units

We consider the following two frameworks to extract chunk units, with Bayesian network as the major framework:

- **Regression-based feature selection or structure learning framework.** Based on regression models, many *efficient* feature selection or structure learning methods already exist. However, the limitations of this approach include: 1) the compensatory relationship among skills is assumed; 2) it's hard to realize the evidence propagation among skills in a probabilistic way; and 3) it doesn't provide the explicit knowledge level of each individual skill. Still, we might be able to use this framework for exploratory analysis or for pre-selection, due to its potential efficiency.
- **BN-based score-and-search framework.** We can employ a search procedure for learning the structure; namely, what chunk units to include. However, if we don't limit the search space, the complexity will grow exponentially. As a result, we propose a greedy search procedural that requires a pre-ranking of the candidates for chunk units. During each iteration, it compares the cost function value of the network with a chunk unit that is newly incorporated with that of the optimal network so far.

To rank chunk units, we use the following general information that should be available across datasets or domains:

- **Frequency information based on skill to problem q-matrix.** Chunk units with higher frequencies, according to the q-matrix, can be considered to be more typical or stable patterns to be modeled.
- **Performance information based on student performance data.** We can employ various strategies, such as giving higher scores to chunk units with larger difference in the estimated difficulty between the current chunk unit and its hardest constituent skill (unit).
- **Natural language processing on the problem (solution) text.** We can consider information such as the textual proximity and semantics that can be obtained by automatic text analysis (or natural language processing).

To further improve the *interpretability*, *robustness* and *generality*, we can also use some domain-specific knowledge to extract more meaningful or typical chunk units. For example, in programming, we can use the abstract syntax tree as in [15]. However, there are still two other challenges:

- **Model run-time complexity.** Since the network involves latent variables, we use Expectation-Maximization, which computes the posteriors of latent variables in each iteration, which can be a time-consuming process.
- **Temporal learning effect.** It is also challenging to consider the temporal learning effect in such a complex network. As a first step, we ignore it during the model learning process, while maintaining the dynamic knowledge estimation during the application phase, as in [3].

We expect to explore some efficient implementations and techniques (such as re-using some posteriors or using approximate inference) to address these two challenges.

3.2 Model Evaluation

We will conduct both data-driven and classroom study evaluations to compare our model with alternatives, including traditional KT-based models [4, 5], and BN-based models with chunk units incorporated in a non-hierarchical way.

3.2.1 Data-driven Evaluation

First, we will conduct data-driven evaluations that consider:

- **Mastery accuracy and effort**. The basic idea of the mastery accuracy metric is that once a student model asserts mastery for an item's required skills, the student should be very unlikely to fail the current item. Meanwhile, the mastery effort metric empirically quantifies the number of practices that are needed to reach mastery of a set of skills. These metrics extend our approach in [6].
- **Parameter plausibility.** This metric investigates how much the fitted parameters can satisfy a model's assumptions and can be interpreted by a human. This is based on our recent Polygon evaluation framework [9].

- **Predictive accuracy of student answers.** This metric evaluates how well the new model predicts the correctness of a student's answer, or the content of a student's solution, based on the problem type.

3.2.2 Classroom study evaluation

We will conduct classroom studies, based on an adaptive learning system that applies our new student model. This system will contain a new open student model interface and a new recommendation engine that will be enabled by our new student model. We will focus on following questions:

1. Do students agree more with the knowledge and mastery inference obtained from the new student model?
2. Does the new student model increase students' awareness of pursuing true mastery?
3. Does the new student model enable more helpful recommendation or remediation?
4. Do students using the new adaptive learning system enabled by the new student model achieve deeper learning which is measured by specifically designed tests?

4. CURRENT WORK

We have conducted preliminary studies with skill chunk units extracted from pairwise skill combinations on a Java programming comprehension dataset and a SQL generation dataset collected across two years from University of Pittsburgh classes. Due to the runtime limitation, we employed a heuristic approach to choose skill combinations (without a complete search procedural), and conducted data-driven evaluations (by 10-fold cross validation). We found that incorporating pairwise skill combinations can significantly increase mastery accuracy and more reasonably direct students' practice efforts, compared to traditional Knowledge Tracing models and its non-hierarchical counterparts. The details of this study are reported in [10].

5. ADVICE FOR FUTURE WORK

I am seeking advice on any of the following aspects:

1. Is this idea both significant and valuable? For example, can it be connected or applied in a broad range of tutoring systems or domains?
2. Are there any datasets, domains or tutoring systems suitable for exploring this idea? What should be the desirable characteristics of the datasets?
3. Are there better representations for skill chunks within or beyond Bayesian networks (e.g., Markov random field, case-base reasoning)? Are there better techniques to identify such units?
4. Are there any suggestions for the overall procedures of this research? For example, should we do a user study to investigate this phenomenon before data mining? If so, how should we design such a study, since we can only test limited chunk units? Should we construct ideal datasets where chunk units are expected to be significant, rather than focusing on existing datasets?
5. How should we situate our definition of chunk units in a broader context considering different domains, problem (task) types and cognitive psychology theories? Is *chunk* the right word? What's its connection with production rules, declarative and procedural knowledge, Bloom's taxonomy?

6. REFERENCES

[1] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis: A general method for cognitive model evaluation and improvement. In M. Ikeda, K. Ashley, and T.-W. Chan, editors, *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 164–175. Springer Berlin / Heidelberg, 2006.

[2] W. G. Chase and H. A. Simon. Perception in chess. *Cognitive psychology*, 4(1):55–81, 1973.

[3] C. Conati, A. Gertner, and K. Vanlehn. Using bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, 12(4):371–417, 2002.

[4] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1995.

[5] Y. Gong, J. E. Beck, and N. T. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *Proc. 10th Int. Conf. Intelligent Tutoring Systems*, pages 35–44. Springer, 2010.

[6] J. P. González-Brenes and Y. Huang. Your model is predictive but is it useful? theoretical and empirical considerations of a new paradigm for adaptive tutoring evaluation. In *Proc. 8th Intl. Conf. Educational Data Mining*, pages 187–194, 2015.

[7] J. P. González-Brenes, Y. Huang, and P. Brusilovsky. General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. In *Proc. 7th Int. Conf. Educational Data Mining*, pages 84–91, 2014.

[8] N. T. Heffernan and K. R. Koedinger. The composition effect in symbolizing: The role of symbol production vs. text comprehension. In *Proc. 19th Annual Conf. Cognitive Science Society*, pages 307–312.

[9] Y. Huang, J. P. González-Brenes, R. Kumar, and P. Brusilovsky. A framework for multifaceted evaluation of student models. In *Proc. 8th Int. Conf. Educational Data Mining*, pages 203–210, 2015.

[10] Y. Huang, J. Guerra, and P. Brusilovsky. Modeling skill combination patterns for deeper knowledge tracing. In *the 6th Int. Workshop on Personalization Approaches in Learning Environments (In Submission)*, 2016.

[11] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5):757–798, 2012.

[12] B. Mostafavi and T. Barnes. Evolution of an intelligent deductive logic tutor using data-driven elements. *International Journal of Artificial Intelligence in Education*, pages 1–32, 2016.

[13] Z. A. Pardos, N. T. Heffernan, B. Anderson, and C. L. Heffernan. The effect of model granularity on student performance prediction using bayesian networks. In *Proc. 11th Int. Conf. User Modeling*, pages 435–439. Springer, 2007.

[14] R. Pelánek et al. Application of time decay functions and the elo system in student modeling. *Proc. 7th Int. Conf. Educational Data Mining*, pages 21–27, 2014.

[15] K. Rivers and K. R. Koedinger. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *International Journal of Artificial Intelligence in Education*, pages 1–28, 2015.

[16] E. Soloway and K. Ehrlich. Empirical studies of programming knowledge. *IEEE Trans. Software Engineering*, SE-10(5):595–609, 1984.

[17] E. Tulving and F. I. Craik. *The Oxford handbook of memory*. Oxford: Oxford University Press, 2000.

Using Case-Based Reasoning to Automatically Generate High-Quality Feedback for Programming Exercises

Angelo Kyrilov
University of California, Merced
5200 North Lake Road
Merced, CA 95343, USA
akyrilov@ucmerced.edu

ABSTRACT

My research explores methods for automatic generation of high-quality feedback for computer programming exercises. This work is motivated by problems with current automated assessment systems, which usually provide binary ("Correct"/"Incorrect") feedback on programming exercises. Binary feedback is not conducive to student learning, and has also been linked to undesirable consequences, such as plagiarism and disengagement.

We propose a Case-Based Reasoning approach to utilize knowledge created by human instructors in order to automatically generate comparable responses for students that submit incorrect solutions to programming exercises. Such a system would offer significant labor savings for instructors, without sacrificing the quality of student learning.

Preliminary experiments have demonstrated the strength of our Case-Based Reasoning approach and its potential impact, especially in MOOCs. Further research is being conducted in order to refine the procedure and to evaluate it effect on student learning.

1. INTRODUCTION

Computer programming is becoming an essential skill in today's economic climate. This has led to significant enrollment increases for introductory Computer Science (CS) courses, as students from virtually all disciplines are required to learn programming. In order to cope with the increased workload, many CS educators rely on automated assessment systems for programming exercises.

Automated Assessment systems for computer programming exercises have been studied widely. [1] provides an overview of automated assessment approaches, and [7] studied the effectiveness of automated assessment on student learning. The authors found that systems which offer instant feedback and allow for multiple resubmissions are helping students to learn.

Some researchers are opposed to using such systems, mainly because of the poor quality of feedback they offer students. In many cases feedback is limited to a binary response ("Correct"/"Incorrect"). [2, 6] argue that in order for learning to take place, students who have generated incorrect solutions to a particular programming exercise, need to be given *guidance* by an expert programmer, and that simply pointing out the presence of an error is not enough.

We studied the effects of binary feedback on students and found that it increases their propensity to cheat on programming assignments and/or disengage from the course material [4]. A possible explanation for this is that since a binary response does not explain the reasons for failure, nor does it suggest a possible strategy to resolve the problem, students are often left with little choice but to cheat or given up on the exercise.

In [3], we proposed a Case-Based Reasoning approach to address the issues surrounding binary instant feedback. The idea is to use knowledge previously generated by human instructors in order to automatically build meaningful responses to incorrect programs submitted by students. In practice, such a system would have a signifiant impact in both traditional classroom environments as well as Massive Online Open Courses (MOOCs). We believe that automated feedback, comparable in quality to human-generated responses, will address motivation problems in MOOCs, which is expected to lead to increased completion rates. In regular university settings, the labor savings will allow instructors and teaching assistants to spend more time on activities beneficial to their students, rather than grading or debugging students' code.

The rest of the paper is organized as follows. Section 2.1 is an overview of our research, and a motivation for the chosen directions. Section 3 presents preliminary results, and highlights the potential contributions of this work. Section 4 outlines research questions that will be explored in future studies, and Section 5 contains concluding remarks.

2. RESEARCH TOPIC

2.1 Case-Based Reasoning

Case-Based Reasoning (CBR), first introduced by Schank [5], is a problem solving framework that uses past experiences to solve problems. Past experiences, referred to as a *cases*, are stored in a database, known as the *case base*. A single case consists of a problem description and a solution.

When a new problem, or a *query*, is encountered, the CBR system retrieves past cases whose problem descriptions are similar to the new problem, and uses the past solutions to generate instructions on how to solve the query. If executing the instructions does not lead to a solution of the problem, then the instructions are revised and evaluated again. Revisions may take place multiple times, until the solution generated by the system is accepted. At this point a new case, made up of the query and the accepted solution, is stored in the case base, making additional knowledge available for future queries. Due to its ability to create new knowledge in this way, CBR is considered a machine learning technique.

The CBR process can be summarized as t
stages, illustrated graphically in figure 1:

1. **Retrieve:** Retrieve past cases that
query.
2. **Reuse:** The retrieved cases are us
solution to the query.
3. **Revise:** The solution generated in th
uated and modified if necessary.
4.

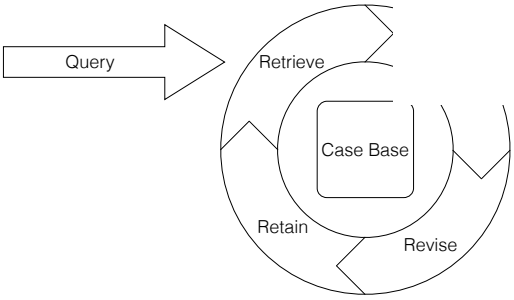


Figure 1: The case-based reasoning methodology

2.2 Proposed System

The automated assessment system we propose utilizes a Case-Based Reasoning approach to automatically assess computer programming exercises and provide feedback to students. We define a case to be a pair made of an incorrect computer program P, and instructor-generated feedback F. A computer program is deemed incorrect if it does not produce the expected outputs for a given programming exercise. Cases are therefore exercise-specific. Our case base is simply a collection of such cases.

For the retrieval stage, we need to define method of computing similarity between cases. Two cases (P_1, F_1), and (P_2, F_2) are said to be similar if P_1 is *similarly incorrect* to P_2. Two programs are similarly incorrect if they both contain the same bugs, therefore corrective feedback for one of the programs is equally appropriate for the other. In the reuse stage, we use the feedback retrieved at the previous step, without any modifications. This is possible due to the way we have defined the similarity metric for two cases.

The revise step, if necessary, will be performed by a human instructor. This is the way the system creates new knowledge. The revise procedure will be invoked if the student repeatedly submits an incorrect solution to the same exercise. This would suggest that the feedback offered by the system has not been helpful to the student. Once a correct solution has been submitted by the student, a new case is stored in the database. This case is made up of the original incorrect source code and the feedback that led to the

phical

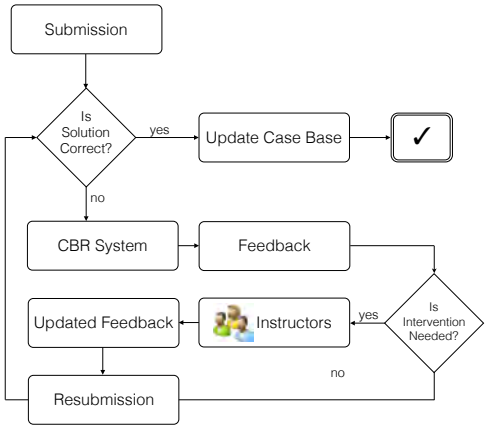


Figure 2: A flowchart of proposed system

2.3 Motivation

Previous research on Case-Based Reasoning has shown that the technique is most effective when similar problems are encountered often and when similar problems have similar solutions. Both of these conditions hold in the context of computer programming exercises. Indeed, CS educators often see the same mistake made by many students, and due to the asynchronous nature of laboratory sessions, the instructor is forced to give the same explanation to multiple students. The second condition, that similar problems have similar solutions holds true as well. If two or more students have all made the same mistake, they will all benefit from the same explanation. There could be multiple ways to explain the same mistake, and some students may find one explanation more beneficial than others. This is easily addressed by allowing the system to store multiple feedback comments per case, and present them sequentially upon unsuccessful attempts. The system can also keep track of the likelihood a particular feedback comment will lead to a successful resubmission and use this information to determine the order in which comments will be presented. Both conditions have been verified experimentally, with results presented in Section 3.

3. PRELIMINARY RESULTS

To test the soundness of our proposed system, we gathered student submissions from an undergraduate Computer Science course where students were required to complete pro-

gramming exercises on a weekly basis. Students uploaded their solutions to an automated assessment system that evaluated their correctness using unit testing.

The first research question we sought to answer was whether or not our proposed system was feasible. We randomly selected 5 exercises and extracted all the incorrect submissions for each one. We then manually clustered them according to their incorrectness. The results from this clustering procedure are presented in Table 1.

Exercise Number	Incorrect Submissions	Cluster Count	Largest Cluster	Smallest Cluster
1	111	4	54	2
2	82	10	18	1
3	73	11	19	1
4	28	8	15	1
5	26	8	13	1

Table 1: Summary of clustering experiment

It is clear from Table 1 that the same mistakes are made by many different students. This is indicated by the large values in the "Largest Cluster" column. In 4 of the 5 exercises we considered, there were mistakes committed by only one student, but small clusters are generally rare.

A more interesting and significant finding was that the number of clusters is relatively small compared to the total number of incorrect submissions. The average number of clusters is 8. This means that there are only 8 different mistakes that students are making, on average. This result is significant because an instructor with an empty case base will only need to grade 8 exercises by hand. The CBR system would be able to provide the appropriate feedback to every subsequent incorrect submission. The number of clusters is also not expected to grow with the number of students enrolled in the class. This is because the number of clusters is a function of the problem, not the number of students.

If the system scales well, it would enable MOOC instructors to provide corrective feedback to tens of thousands of students who have submitted incorrect solutions to programming exercises. This is likely to increase student engagement with the material and improve overall completion rates.

4. FUTURE WORK

In order to realize our system design, we need a reliable way to automatically detect similarity with respect to incorrectness between two programs. Our initial approach was to compute this similarity based on the unit tests. That is if two programs fail the exact same set of unit tests then they are deemed to be similarly incorrect. This is a reasonable first approach but it generates many false positives and false negatives. To ensure true scalability, the false matches need to be kept to a minimum. Methods involving static analysis of source code will likely need to be employed.

Further investigation of our scalability claims is also needed. More submission data would have to be analyzed and relationships between class size and number of clusters would need to be formally established.

Once the system has been completed, it should be deployed in a classroom and its effectiveness should be studied.

5. CONCLUSION

My research is focused on improving the quality of instant feedback generated by automated assessment systems for programming exercises. Many instructors are using automatic grading systems that are limited to providing binary feedback, which has been shown to hinder student learning and lead to plagiarism and disengagement.

We propose a Case-Based Reasoning approach to designing an automated assessment system for programming exercises capable of instantly delivering high-quality feedback, comparable to guidance a human instructor might provide to a struggling student. The system uses feedback previously generated by human instructors and delivers it to students who make similar mistakes to ones seen before.

This is an effective technique since the same mistakes are made by many different students and there are relatively few distinct mistakes. This translates into significant labor savings for instructors and teaching assistants. With our system in place, an instructor will only have to address a specific problem once. All subsequent occurrences will be handled automatically by the system.

Further research is currently being conducted on finding a reliable metric for similarity with respect to incorrectness of computer programs. Several static analysis techniques are being explored. Attempts are also being made to formalize relationships between the class size and the number of unique errors that can be made on an exercise. We postulate that for reasonably sized programming exercises, the number of unique errors will stay low even in MOOC environments where class sizes can be in the hundreds of thousands.

6. REFERENCES

[1] K. M. Ala-Mutka. A survey of automated assessment approaches for programming assignments. *Computer science education*, 15(2):83–102, 2005.

[2] T. Beaubouef and J. Mason. Why the high attrition rate for computer science students: Some thoughts and observations. *SIGCSE Bull.*, 37(2):103–106, June 2005.

[3] A. Kyrilov and D. C. Noelle. Using case-based reasoning to improve the quality of feedback provided by automated grading systems. In *Proceedings of the International Conference on E-Learning*, pages 384–388, 2014.

[4] A. Kyrilov and D. C. Noelle. Binary instant feedback on programming exercises can reduce student engagement and promote cheating. In *Proceedings of the 15th Koli Calling Conference on Computing Education Research*, Koli Calling '15, pages 122–126, New York, NY, USA, 2015. ACM.

[5] R. Schank. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. Cambridge University Press, New York, NY, USA, 1982.

[6] G. N. Walker. Experimentation in the computer programming lab. *Inroads*, 36(4):69–72, 2004.

[7] D. Woit and D. Mason. Effectiveness of online assessment. *SIGCSE Bull.*, 35(1):137–141, Jan. 2003.

Predicting Off-task Behaviors in an Adaptive Vocabulary Learning System

SungJin Nam
School of Information
University of Michigan
Ann Arbor, MI 48109
sjnam@umich.edu

ABSTRACT

In many studies, engagement has been considered as an important aspect of effective learning. Retaining student engagement is thus an important goal in intelligent tutoring systems (ITS). My current studies with collaborators on Dynamic Support of Contextual Vocabulary Acquisition for Reading (DSCoVAR) include building prediction models for students' off-task behaviors. By extracting linguistically meaningful features and historical context information from interaction log data, these studies illustrate how some types of off-task behavior can be modeled from behavioral logs. The results of this research contribute to existing studies by providing examples of how to extract behavioral measures and predict off-task behaviors within a vocabulary learning system. Identifying off-task behaviors can improve students' learning by providing personalized learning materials: for example, off-task behavior classifiers can be used to achieve more accurate predictions of the student's vocabulary mastery level, which in turn can improve the system's adaptive performance. Toward our goal of developing highly effective personalized vocabulary learning systems, this research would benefit from expert feedback on issues that include: principled approaches for adaptive assessment and feedback in a vocabulary learning system; and alternative methods for defining and generating off-task labels.

Keywords

Engagement, off-task behaviors, prediction model, log data, intelligent tutoring system, adaptive system

1. INTRODUCTION

Engagement has long been considered as an important aspect of learning [17, 16]. Engagement is a comprehensive behavior that reflects an integration of different aspects of a person's cognitive state [11, 6, 7]. A student's engagement level while using the system can vary with time, and it can be influenced by many factors, such as the difficulty of questions, prior experience with similar technology, and individual interests or motivation [14, 1]. Thus, measures related to engagement need to consider the multidimensional construct of engagement and clarify which types of engagement are going to be measured in the study [18].

Other studies based on digital learning environments tend to capture engagement based on behavioral signals. Studies on intelligent tutoring systems (ITS) often used features like response time, number of erroneous attempts, and frequent accessing of hint messages to predict students' engagement [2, 4]. Studies in Massive Online Open Courses (MOOC) included features like the number of lecture videos seen, participation in pop-up quizzes, and social interactions like frequency of article posting or comments in the discussion forum, to predict the student's overall participation level [10, 15]. These studies showed that data traces of observable behavior can be used to predict student engagement, often operationalized as a classroom attitude observed from instructors or a survival rate of enlisted courses in a MOOC.

The purpose of this research topic is to model a particular subset of students' off-task behaviors while they use a vocabulary learning system, based on observations of their interaction from log data. In our study, each student response to an assessment question posed by the system was defined as an off-task behavior if it contained less serious, patterned, or repetitive errors [13, 12]. Key research questions on this topic that I will explore include: (1) identifying important predictive features of off-task behaviors in vocabulary learning systems that can be collected from log data, (2) evaluating different modeling methods that can help to develop more accurate prediction models for off-task behaviors, and (3) suggesting effective adaptive strategies for vocabulary learning systems that will help to sustain student's engagement and thus improve their learning outcomes and experience. The results from our current studies are expected to be used maximize the efficiency and long-term effectiveness of student learning outcomes.

2. CURRENT WORK AND RESULTS

Currently, I am working on developing a contextual word learning (CWL) system called Dynamic Support of Contextual Vocabulary Acquisition for Reading (DSCoVAR)[1]. DSCoVAR is an online vocabulary learning system that teaches K-12 students how to figure out the meaning of a word they don't know (sometimes called the *target word*) by using clues from the target word's surrounding context[8].

The DSCoVAR curriculum consists of three sessions: pre-

[1]http://dscovar.org

test, training, and post-test sessions. Questions in the pre- and post-test sessions include multiple types of questions measuring the student's knowledge on vocabulary before and after the training session. The training session consists of an instructional video and practice questions that teach the student different strategies for figuring out the meaning of an unknown target word by using clues from nearby words in the surrounding sentence. Students learned, and were tested on, a family of words known as Tier-2 words, which are words that are critical for understanding more advanced texts, but that are relatively rare in everyday use. These target words were expected to be difficult, but at least familiar or known to a small number of students. (In our first experiment, participants reported that they were Familiar with 26% of the Tier 2 target words, followed by 21% Known, and 53% Unknown (N=33) [13].)

2.1 Feature Extraction

In previous studies [13, 12], we analyzed students' responses in the pretest session and developed prediction models for off-task behaviors based on behavioral features extracted from log data. During sessions, DSCoVAR recorded how students interacted with the system by storing time-stamped event data and students' text responses. Based on the collected log data, we extracted two types of variables: response-time variables (RTV) and context-based variables (CTV). These variables contain more meaningful student behavior information than the raw log data, and are used as predictor variables in our off-task behavior classifiers.

RTVs collect information right after the student submits his or her response for each question, including time spent to initiate and finish typing a response, the number of spelling and response formatting errors, and orthographic and semantic similarity between the response and the target word. CTVs include history-based measures relating to how the student performed in previous trials (with different window sizes of 1, 3, 5, and 7), such as the average proportion of off-task responses in previous trials and average orthographic or semantic overlap between the current response and previous responses. Lastly, human raters created labels for off-task behaviors from log data. By using criteria based on Baker et al. [3], we obtain labels for certain types of off-task behavior, i.e. when responses seemed less serious and patterned, or when they involved repetitive errors.

2.2 Modeling Off-task Behaviors

With the RTVs and CTV features described above, we build off-task prediction models via mixed effect models and structure learning algorithms. Mixed effect models, such as the generalized linear mixed effect model (GLMM) or hierarchical Bayesian model, are suitable for analyzing the log data from ITS since they can account for variance across repeated measures like multiple responses from a single student or a particular target word.

Table 1 and 2 show the results of the GLMM model learned by the stepwise algorithm for predicting the off-task labels from RTV and CTV variables. GLMM includes random intercepts for target words and students, and the effect of random slopes for the student's prior familiarity level to the target word mentioned above 2 [13, 12]. The results show that RTV features like response length and orthographic similarity between the response and the target word are statistically significant for explaining the specific types of off-task behavior that we identified for the study. CTVs like average semantic similarity between the current response and previous three responses and orthographic similarities with previous seven responses were also significant. This model showed a better area under the curve statistic from ROC curve (0.970) than the RTV-only GLMM model (0.918).

Table 1: GLMM results for fixed effect variables (all predictors are statistically significant ($p < 0.001$))

Variables	Coeff	SE	z
(Intercept)	0.50	0.62	0.82
RTV: Response Length	-0.22	0.05	-4.10
RTV: Ort. Similarity	-5.98	1.79	-3.34
CTV: Sem. Similarity (prev. 3)	0.11	0.03	4.35
CTV: Ort. Similarity (prev. 7)	11.4	1.81	6.33

Table 2: GLMM results for random effect variables

Variables	Var.	Corr.
Target (Intercept)	1.05	
Target-Unknown:Known	2.47	-1.00
Target-Unknown:Familiar	23.0	-1.00
Subject (Intercept)	3.67	

Structure learning algorithms, such as the stepwise regression and the Hill-climbing algorithm, were used for automatically learning the model structure of off-task prediction models. The stepwise algorithm was useful in selecting which variables can bring the better fit to the regression model based on criteria like AIC or BIC. The Hill-climbing algorithm was helpful for identifying the complex interaction structures between variables based on conditional probabilities. By combining findings from different structure learning algorithms, we confirmed that adding interaction structures is helpful for prediction, especially with RTV-only models. An example of interaction structures learned from the Hill-climbing algorithm is shown in Figure 1.

3. PROPOSED CONTRIBUTIONS

First, the current work contributes to existing ITS studies by suggesting methods for extracting meaningful information from log data. For example, RTVs provided meaningful information to understand student performance on specific questions by using various language processing techniques, such as orthographic similarities measured using character trigrams, and semantic similarities measured using Markov Estimation of Semantic Association [9]. CTVs provided information on historical patterns of off-task behaviors. Combined with mixed effect models, our results suggest that traditional predictive features, such as time spent for initiating and finishing the response or number of error messages, can be substituted (when available) with features based on variance in repeated measures and contextual information.

Second, identifying off-task status at the item level can be a starting point for managing student engagement systematically, by letting the learning system know when to intervene in helping the student regain their engagement to the task. Off-task classifiers in the current studies provided examples of automatized models for checking student engagement in a vocabulary learning system.

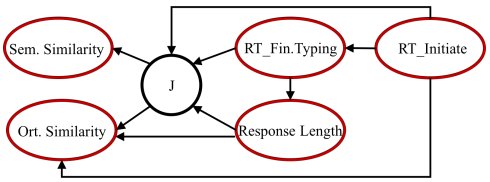


Figure 1: Interaction structure of RTVs learned by the Hill-climbing algorithm (Node J: Off-task label)

Third, this research can be helpful for achieving more accurate predictions on the student's vocabulary mastery level. For example, suggested classifiers provide item-level prediction for off-task behaviors based on previous responses. These results can be helpful for distinguishing between intentionally missed questions and accidentally erroneous responses, which in turn can be used to improve estimates provided by existing student learning prediction models, such as item response theory [5].

4. FUTURE DIRECTIONS

A key goal of this research is to build an adaptive vocabulary learning system. By using results from our current studies, we will implement an initial adaptive mechanism in DSCoVAR that personalizes the difficulty of training session's questions based on a student's estimated vocabulary mastery. This approach is expected to help retain student engagement with the system by providing the right level of 'desirable difficulty' while also making more efficient use of the student's learning time. However, it is unclear how features related to perceived question difficulty, such as amount of information given from feedback messages or size of spacing between questions that share the same target, could be used to model the overall student engagement with the question. Advice from experienced researchers on adaptively controlling task difficulty would help guide this research on personalized training to students.

Our current work depends on defining a specific type of off-task behavior, with labels generated from two human judges. While the inter-rater agreement was reasonable (Cohen's Kappa of 0.695) [12], it is an expensive process and the number of collectible judgments are limited. An alternative approach could be to use crowd-sourcing for labeling the log data. However, converting this expert labeling task into a fragmentary job for anonymous workers may require more carefully designed instructions and robust methods for validating the credibility of labels. Expert guidance on alternate definitions of off-task behavior, and improved approaches for gathering larger amounts of labeled data based on these definitions, would be helpful for expanding future studies.

5. ACKNOWLEDGMENTS

This research was supported in part by the Institute for Educational Sciences (IES R305A140647) through a grant to the University of Michigan. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education. I also thank Dr. Kevyn Collins-Thompson and Dr. Gwen Frishkoff for their guidance and suggestions.

6. REFERENCES

[1] I. Arapakis, M. Lalmas, B. B. Cambazoglu, M.-C. Marcos, and J. M. Jose. User engagement in online news: Under the scope of sentiment, interest, affect, and gaze. *Journal of the Association for Information Science and Technology*, 65(10):1988–2005, 2014.

[2] I. Arroyo and B. P. Woolf. Inferring learning and attitudes from a bayesian network of log file data. In *AIED*, pages 33–40, 2005.

[3] R. S. Baker, A. T. Corbett, and K. R. Koedinger. Detecting student misuse of intelligent tutoring systems. In *Intelligent tutoring systems*, pages 531–540. Springer, 2004.

[4] J. E. Beck. Engagement tracing: using response times to model student disengagement. *Artificial Intelligence in Education: Supporting Learning Through Intelligent and Socially Informed Technology*, 125:88, 2005.

[5] S. E. Embretson and S. P. Reise. *Item response theory for psychologists*. Psychology Press, 2000.

[6] J. A. Fredricks, P. C. Blumenfeld, and A. H. Paris. School engagement: Potential of the concept, state of the evidence. *Review of educational research*, 74(1):59–109, 2004.

[7] J. A. Fredricks and W. McColskey. The measurement of student engagement: A comparative analysis of various methods and student self-report instruments. In *Handbook of research on student engagement*, pages 763–782. Springer, 2012.

[8] G. Frishkoff, K. Collins-Thompson, and S. Nam. Dynamic support of contextual vocabulary acquisition for reading: An intelligent tutoring system for contextual word learning. In *Adaptive Educational Technologies for Literacy Instruction*. Taylor & Francis, Routledge:NY, In Press.

[9] G. A. Frishkoff, K. Collins-Thompson, C. A. Perfetti, and J. Callan. Measuring incremental changes in word knowledge: Experimental validation and implications for learning and assessment. *Behavior Research Methods*, 40(4):907–925, 2008.

[10] R. F. Kizilcec, C. Piech, and E. Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, pages 170–179. ACM, 2013.

[11] K. R. Koedinger, E. Brunskill, R. S. Baker, E. A. McLaughlin, and J. Stamper. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, 34(3):27–41, 2013.

[12] S. Nam, K. Collins-Thompson, and G. Frishkoff. Modeling real-time performance on a meaning-generation task. In *Annual Meeting of the American Educational Research Association*. AERA, 2016.

[13] S. Nam, K. Collins-Thompson, G. Frishkoff, and L. Hodges. Measuring real-time student engagement in contextual word learning. In *The 22nd Annual Meeting of the Society for the Scientific Study of Reading*. SSSR, 2015. `https://goo.gl/CvTL1K`.

[14] H. L. O'Brien and E. G. Toms. What is user engagement? a conceptual framework for defining user engagement with technology. *Journal of the American Society for Information Science and Technology*, 59(6):938–955, 2008.

[15] A. Ramesh, D. Goldwasser, B. Huang, H. Daume III, and L. Getoor. Learning latent engagement patterns of students in online courses. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[16] B. Ravindran, B. A. Greene, and T. K. Debacker. Predicting preservice teachers' cognitive engagement with goals and epistemological beliefs. *The Journal of Educational Research*, 98(4):222–233, 2005.

[17] J. P. Rowe, L. R. Shores, B. W. Mott, and J. C. Lester. Integrating learning and engagement in narrative-centered learning environments. In *Intelligent Tutoring Systems*, pages 166–177. Springer, 2010.

[18] G. M. Sinatra, B. C. Heddy, and D. Lombardi. The challenges of defining and measuring student engagement in science. *Educational Psychologist*, 50(1):1–13, 2015.

Estimation of prerequisite skills model from large scale assessment data using semantic data mining

Bruno Elias Penteado
ICMC - Institute of Mathematics and Computer Science
University of Sao Paulo
Av. Trabalhador São-Carlense, 400, São Carlos, Brazil
brunopenteado@usp.br

ABSTRACT

Learning sequences are important aspects in learning environments. Students should learn by moving gradually from simpler to more complex concepts, promoting deeper levels of learning. This feature is usually embedded in most intelligent learning environments to guide the student in the study of subject matter. The organization of this knowledge structure is usually an intensive effort of human experts, in creating a logical ordering of what is to be taught - determining the concepts and the prerequisite relations among them. In recent years, some methods have been developed for dealing with this knowledge structuring using data coming from logs of learning environments, applying data mining techniques to discover prerequisite rules and create directed graphs of prerequisites. These methods model both assessment items and skills underlying those items. The automatic methods developed so far present a semantic gap between the probabilistic analysis and the expert knowledge, sometimes causing confusion with the results. This research aims to bridge this gap by adding a minimal layer of semantic information to help in the data mining process. As an application, we intend to analyze large-scale assessment datasets, considering its specificities, and evaluate if those hybrid models can improve the prediction of item success.

Keywords

Skill model, knowledge structure, data mining, semantic data mining.

1. INTRODUCTION

Skills prerequisite structure is an important component in *domain modeling*, used in intelligent learning environments and which serve as a basis for planning learning sequences and adaptive strategies for tutoring systems. Analogously, most intelligent learning environments uses a *student model* for the automatic adaptation of teaching strategies and as a overlay of domain model, influencing how the automatic intervention is carried out. Human experts usually define such prerequisite structure; however, they are rarely validated empirically and improved for better results.

For most of the large scale assessments, the current approach considers all knowledge in a single unidimensional scale, which considers the item difficulty in its ordination. *Computer adaptive tests* tend to use predominantly this ordination for item selection in diagnostic assessments. This approach raises some issues: the *interpretability* of results, since a single value is used to represent a knowledge in a large domain; and the *agreement* about the structure, since most experts cannot see a direct, unidimensional relationship among skills. Given the amplitude of skills, experts seem to agree on other sorts of dependencies, not just the simple ordination for item difficulty. For instance, in the field of Physics, an easy item of spatial movement might not be considered as a prerequisite for a difficult item in geometric optics, since they belong to different branches.

On the other hand, the process of manual creation of these dependencies is highly costly, time-consuming and presents large disagreement among experts modeling the same domain. Pavlik et al. [1] point to 3 other factors: the description of irrelevant skills, redundancy among skills and the ordination of those skills

There seems to be a semantic gap between the automatic extraction from data and the mapping made by human experts. This research aims to explore this gap, trying to bridge it using semantic data mining, and combining the advantages of both approaches.

2. PREVIOUS WORK

The process of prerequisite structure derivation from observable variables (such as assessment items) from data has been investigated by many researchers; yet, the skill modeling is still an open issue, since a student's knowledge is a latent variable, not being observed directly. In [2] it is proposed the POKS (Partial Order Knowledge Structure) algorithm to learn the dependency structure among items, composed only by the observable nodes (answers to the items), outperforming Bayesian networks algorithm, both in predictive performance and computational efficiency. In [1] POKS algorithm is applied to analyze the relations among skills, using observable items and use the result to cluster redundant skills, with a high degree of covariance, simplifying the domain model and determining it structure. In [3] a method is proposed to determine dependency relations among curricular units from student's performance data, using a binomial test for every pair of skills, to evaluate the existence of a prerequisite relationship between them. In [4] a frequent association rules mining method is proposed to discover concept maps, but not considering the uncertainty in the process of knowledge transfer of the student to his performance. In [5] the structure is derived from noisy observations using log likelihood calculated between the precondition model and the model in which the skills are all independent on each pair of skills to estimate which model better fits the student's data. In [6] causal discovery algorithms are used to find a skill prerequisite structure applying statistical tests in the latent variables. In [7] is proposed a probabilistic association rules mining method, having the probabilistic knowledge states estimated by an evidence model, to find a structure from performance data.

In semantic technologies, ontologies are explicit specifications of conceptualization and a formal way to define the semantics of knowledge and data. Dou et al. [8] surveys this semantic data mining in multiple domains - formal ontologies have been introduced to semantic data mining to: i) bridge the semantic gap between data, data mining algorithms and results; ii) provide data mining algorithms with a priori knowledge, guiding the mining process or reducing the search space; iii) provide a formal way for representing the data mining flow, from data preprocessing to mining results. Bellandi et al. [9] presented an ontology-based association rule mining method, using the ontology to filter instances in the process, constraining the search space of itemsets, excluding items and characterizing others according to an abstraction level, enabling generalization of an item to a concept of the ontology. Marinica and Guillet [10] presented a post-processing method for the results of the association mining, pruning invalid or inconsistent association rules with the help of the ontology.

Large scale assessments present some specificities: they are very strict in their skill model, with reference matrices specifying what is expected in the test; they are periodic, meaning that they are applied, in some cases, in an annual basis, with no single item in common between applications; the test items are organized in blocks (incomplete balanced blocks) and the test is comprised of a few blocks with a fixed number of items, so that many versions of the test are available at a time; the items are all pre-tested before the actual application, to estimate psychometric parameters (following Item Response Theory principles) being equalized into the same scale. A challenge for this research is to work with datasets from multiple years (i.e., no common items), balanced in blocks trying to discover generalizations in the underlying skill model.

3. METHOD AND MATERIAL

In this work, we will work with microdata from ENEM – an annual Brazilian exam for high school students, used as a classification ranking for admission in many public federal universities in Brazil. This exam is composed by 4 knowledge areas (Mathematics, Natural Sciences, Human Sciences and Languages), each composed by 30 skills in the reference matrix specified for this exam. Each item is mapped to a single skill and a score is given for each of these knowledge areas. The test is composed by 45 multiple-choice items for each knowledge area, along with an essay, in a 2-day time span. Different tests are organized in an *incomplete balanced blocks* design. In this approach, each test is composed by multiple blocks of items, with fixed ordination and in increasing order of difficulty. The blocks are arranged in different tests so to alleviate possible biases like the position of an item and a fatigue factor for items in the end of the test.

The datasets contain every alternative selected by every student whom participated in the exam. We plan to conduct this study using the Mathematics dataset, from 2009 to 2014, in a sum of 270 items answered my tens of millions of students.

Working along with Math experts, we will try to create simple ontologies, just with constraints of what should or not be considered in the final model, to prune some of the spurious results.

This research will adopt a quantitative approach and use data mining techniques as a method to construct the mapping of the prerequisite structure which, from the items mapped to their respective skills and the performance data (correct and incorrect answers) for every respondent, is able to extract relations among the skills, generalized by different observations in different items.

The evaluation of the method will be based on the capacity of prediction of success on the items individually, assessing the goodness of fit against the human experts mapping. The method will be compared to state-of-the-art algorithms such as POKS, probabilistic association rules mining and with some expert mapping.

4. PRELIMINARY WORK

This is a research project in its earlier stages, narrowing the research questions to be pursued. As an initial effort, I found that more simplistic approaches tend to model just the difficulty of items in the creation of a prerequisite structure, i.e., an easier item is a prerequisite for a more difficult item, disregarding contextual information on the respective topics.

Early examples for ENEM using data from Mathematics test applied in 2014 are depicted in Figure 1 (skill prerequisites). They were generated by the author using the POKS algorithm, with source code available in [11] and show the algorithm results.

In Figure 1, the previous items were mapped to their respective skills and the algorithm was run. Skills are numbered according to the official codes available at ENEM website. We can see that some skills are more fundamental, specially numbers 1, 3, 4 and 17. Skills 12, 15 and 22 were not assessed in this test.

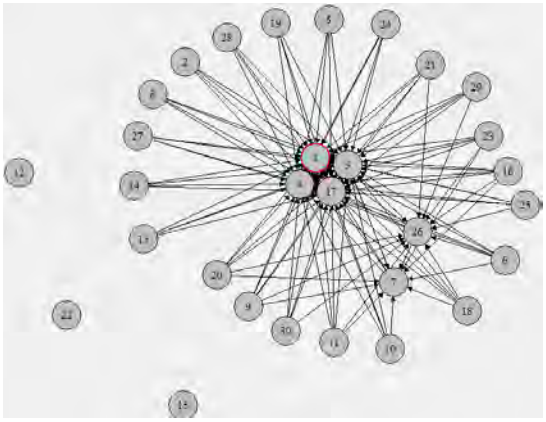


Figure 1. Prerequisite skills derived from Math assessment.

We hope, by the end of this research, discover possible prerequisite relations among skills that constitute the ENEM exam, complementing the traditional model of ordination by item difficulty in the IRT model, by creating a generalized graph of dependencies among skills, estimated from empirical data of application and combined with ontology constraints.

From this mapping, it should be possible to build an intelligent learning environment that might diagnose in which point of the graph the student is and the possible sequences he can choose to study. Another practical implication may be the interpretation of results and extension to practices in public policies. As this sort of exam is applied in different moments in K-12, the model could generalize and describe how learning happens in public education system, since literacy through high school.

5. ADVICES SOUGHT

For this doctoral consortium, advice is sought regarding some concerns:

a) *What data mining methods should be used to model these prerequisite skills?* At first, POKS was used but other methods could also be evaluated, like LFA, Rule Space and BKT. As this is a high stake exam, the skills are wider, different from other more granular skill models from ITS domains. An example (skill 17, a basic skill from Figure 1): "analyze information involving variations in quantity as a resource for argument construction". In addition, the same skill can vary a lot depending on the items being assessed. Second, items being that different and having different difficulty parameter,

b) *Should difficulty be embedded in the model?* so that different items of a same skill can influence differently in the model.

c) *Should these information be included in the model?* which may result in different graphs for different populations. Besides the standard item accuracy prediction. This dataset has no other interaction data, as in ITS systems, but has contextual data about the respondents, with high impact features in performance, like geographic region and socioeconomic status.

d) *Is it valid to measure a interrater agreement metric (like Kappa) to compare the generated model with those from experts?* as a means of comparing how close the model fit the expert modeling.

6. REFERENCES

[1] Pavlik Jr., P.I., Cen, H., Wu, L., Koedinger, K.R.: Using Item-type Performance Covariance to Improve the Skill Model of an Existing Tutor. In Proceedings of the 1st International Conference on Educational Data Mining, Montreal, Canada, 77-86, 2008.

[2] Desmarais, M.C., Meshkinfam, P., Gagnon, M.: Learned Student Models with Item to Item Knowledge Structures. User Modeling and User-adapted Interaction, 16(5), 403-434, 2006.

[3] Vuong, A., Nixon, T., Towle, B.: A Method for Finding Vuong, A., Nixon, T., Towle, B.: A Method for Finding Prerequisites within a Curriculum. In Proceedings of the 4th International Conference on Educational Data Mining, Eindhoven, Netherlands, 211-216, 2011.

[4] Tseng, S.S., Sue, P.C., Su, J.M., Weng, J.F., Tsai, W.N.: A New Approach for Constructing the Concept Map. Computers & Education, 49(3), 691-707, 2007.

[5] Brunskill, E.: Estimating Prerequisite Structure from Noisy Data. In Proceedings of the 4th International Conference on Educational Data Mining, Eindhoven, Netherlands, 217-222, 2011.

[6] Scheines, R., Silver, E., Goldin, I.: Discovering Prerequisite Relationships among Knowledge Components. In Prerequisites within a Curriculum. In Proceedings of the 4th International Conference on Educational Data Mining, Eindhoven, Netherlands, 211-216, 2011.

[7] Chen, Y., Wuillemin, P. H., Labat, J. M. Discovering prerequisite structure of skills through probabilistic association rules mining. In Proceedings of the 8th International Conference on Educational Data Mining, Montreal, Canada, 77-86, 2015.

[8] Dou, D., Wang, H., Liu, H. Semantic Data Mining: a survey of ontology-based approaches. In Proceedings of the 9th International Conference on Semantic Computing, Anaheim, USA, 244-251, 2015. DOI: 10.1109/ICOSC.2015.7050814.

[9] Bellandi, A., Furletti, B., Grossi, V., Romei, A. Ontology-driven association rule extraction: A case study. Contexts and Ontologies Representation and Reasoning, page 10, 2007.

[10] Marinica, C., Guillet, F. Knowledge-based interactive postmining of association rules using ontologies. Knowledge and Data Engineering, IEEE Transactions on, 22(6):784–797, 2010

[11] Desmarais, M., Bhatnager, S. Prerequisite skill structures in ASSISTments. Available in: https://github.com/sameerbhatnagar/POKS-skills. Accessed: March, 20th, 2016.

Designing Interactive and Personalized Concept Mapping Learning Environments

Shang Wang
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe AZ, USA
swang158@asu.edu

ABSTRACT

Concept mapping is a tool to represent interrelationships among concepts. Relevant research has consistently shown the positive impacts of concept mapping on students' meaningful learning. However, concerns have been raised that concept mapping can be time consuming and may impose a high cognitive load on students. To alleviate these concerns, research has explored facilitating concept map construction by presenting students with incomplete templates and concept map based navigational assistance on the learning material. However, it's not clear how these incomplete templates should be designed to address individual student needs and how concept map-based navigation can support students in creating concept maps and developing personalized navigation patterns. In this paper, I discuss my previous research in providing personalized scaffolding in concept mapping activities and describe plans of my research in exploring how personalized concept map scaffolding supported by navigational assistance could enhance student learning.

Keywords

Data mining, concept mapping, navigation, personalization, adaptive scaffolding, expert skeleton concept map.

1. Research Topic

Concept maps are graphical representations of knowledge structures, where labeled nodes denote concepts and links represent relationships among concepts. Concept mapping has been widely employed in educational settings to support student learning. Research has examined how concept mapping tools assist students in summarizing, relating, and organizing concepts [1][4]. However, there are limitations in using concept mapping. The main disadvantage of concept mapping is that the map construction is time-consuming and it requires some expertise to learn [3]. In addition, the complexity of the task often imposes high cognitive load and reduces student motivation [10].

Cañas and colleagues developed CmapTools, a computer-based concept mapping system, to support concept mapping by making it easier to construct and manage large representations for complex knowledge structures [6]. Although CmapTools provides a convenient platform for concept map construction, the system is independent from the learning content and students may encounter difficulties relating maps with resources and comparing linked concepts. To enhance concept maps with relevant resources, McClellan and colleagues designed a system that attaches resources like demos, homework and tutorials to the concept maps via keyword matching [11]. However, it might cause extraneous effort for students to process this additional information.

Apart from providing computer systems for concept map construction, other research canvassed the effect of providing students with incomplete templates called expert skeleton maps, within which some nodes and links were set as blanks, as a scaffolding aid [5]. Although studies show that the scaffolding had more positive effects on student learning than those who created concept maps from scratch [3], it's not clear how expert skeleton maps should be designed to provide better learning results. Questions like what concept nodes should be presented and what concept nodes should be left blank, how big should the expert skeleton map be, and should all students be given the same expert skeleton map, still remain unsolved. To address these challenges and the opportunities from the two directions discussed above, I propose a design of a personalized and interactive concept mapping learning environment that integrates a textbook with a concept mapping tool. This system will enable students to create maps directly from the textbook. Students will relate the created maps to the textbook content and the system will offer personalized scaffolding to facilitate concept map construction and meaningful learning. I also describe my plan of conducting an Amazon Mechanical Turk Study and an in-classroom study to test the system.

2. Proposed Contribution

2.1 Previous Work

Towards designing a personalized and interactive concept mapping learning environment, my prior work has examined how personalized expert skeleton maps affect student learning. More specifically, I studied the potential effects of an adaptive expert skeleton scaffold that contains concepts and relationships for which the student has demonstrated prior knowledge [7]. To create the adaptive expert skeleton maps, an expert concept map representing the knowledge structure from the chapter was first created as a foundation. I then mapped each question on the pretest to a certain part of the expert map to modify the expert skeleton map based on students' pretests scores. For example, if a student incorrectly answered question 4 as shown in Figure 1, the correct concept ("flower") was replaced with '???" and left open for the student to fill in. By presenting students with a map that contained their prior knowledge, I hypothesized that students would spend more effort on unknown concepts and be better supported in integrating new knowledge into prior existing knowledge structure, thus improving learning.

Figure 1. Modifying the expert map based on pre test answers.

To test my hypothesis, I conducted a study with 38 non-biology major students who were randomly assigned into three conditions: (1) adaptive scaffolding, (2) fixed scaffolding and (3) non-scaffolding. Students in the adaptive scaffolding condition received an expert skeleton map that contained nodes and links which they got correct in the pretest. Students in the fixed scaffolding condition also received a skeleton map. However, instead of tailoring the map to the student's prior knowledge, I presented them with maps from the adaptive scaffolding condition through yoked control. In this way, I was able to control for content across conditions. Finally, in the non-scaffolding condition, students constructed a map from scratch. Although I did not discover significant differences in learning gains between conditions, I found that different types of nodes in the template did lead to different learning gains on related concepts. To further investigate, I coded the key ideas in the expert map as being: added to the map by the student, already in the template, or not added. For the already existing concepts in the expert map, I further categorized the concepts that were adjacent to the newly added concepts as "close" and the ones which were more than one link away as "far". Results indicate that students benefit most from adding concept nodes to the map and benefiting more from the in template "close" nodes than the "far" and not added ones.

However, there were several limitations in the data collection that might have influenced the results. First, the number of graduate students and undergraduates was not balanced across conditions, and the learning differences in these two populations may have added extraneous noise to the learning gains. Another potential problem was that the expert skeleton maps students received might have been too large. While I assessed students on 9 key ideas, these ideas spanned more than 70 nodes in our expert map. The complexity of the given template might have imposed high cognitive load on students, reducing the benefits of the expert skeleton maps. What's more, the concept mapping system used in the study, the CmapTools, required students to type the words to create nodes. The system was also limited in terms of searching and comparing resources related to the concept maps.

2.2 System Design

Incorporating my finding discussed above, I present an iPad-based interactive concept mapping tool that is integrated with a digital textbook. When held in landscape mode, the screen splits into two, with the left side displaying the textbook and the right side showing the concept mapping panel. The built-in concept mapping view is directly associated with the learning material, so that students can construct concept maps directly from the words in the textbook, shown in Figure 2. An example of this process would be a student reads the textbook and find the concept "seed" that should be contained in the concept map, he can long click on the word and tap on the add concept button to add a node named "seed" to the concept mapping panel on the right. He can click on other concepts to add and delete links. This feature eliminates the tedious process of manually adding nodes and typing all the text while encouraging the cognitively beneficial processes of finding the important concepts and identifying the relations among them. Apart from that, a hyperlink between the node in the concept map and the words in textbook is created through the "click and add" action, allowing students to navigate through the textbook by clicking on the concept nodes. During their navigation, related concepts in the concept map and the text in the textbook are both highlighted, providing a visual comparison of key information. Since the concept maps are created by students themselves, the system enables students to form their own navigation patterns to assist them in locating key information in the textbook resource. The system is able to provide pre and post tests, which can be used to dynamically modify the expert skeleton map based on the student's prior knowledge. Furthermore, leveraging the hyperlinking navigation feature, the system enables students to click on the nodes in the expert skeleton to navigate directly to related pages.

I hypothesize that the system can alleviate the challenges discussed in the introduction and benefit students in different ways. It first allows students to easily construct concept maps via the "click and add" feature, which reduces the work of tediously typing words into the nodes while preserving the beneficial work of searching and identifying concepts to be added. The hyperlinking navigation provides more flexibility in comparing and finding connections between concepts that are located in different pages. Hyperlinking the expert skeleton map with the textbook enables students to click on the nodes provided in the expert skeleton map to see where these concepts are mentioned in the textbook. This would reduce the cognitive load of the template, which is a potential cause of reducing the effect of adaptive scaffolding in my previous work. What's more, providing students with expert skeleton maps that contain their prior knowledge would facilitate meaningful learning while they add new concept nodes to templates that represent their own knowledge structures. Since the concept nodes are already mastered by students, this approach also avoids potential shallow learning, which is a problem faced by many forms of computer-based instruction [8].

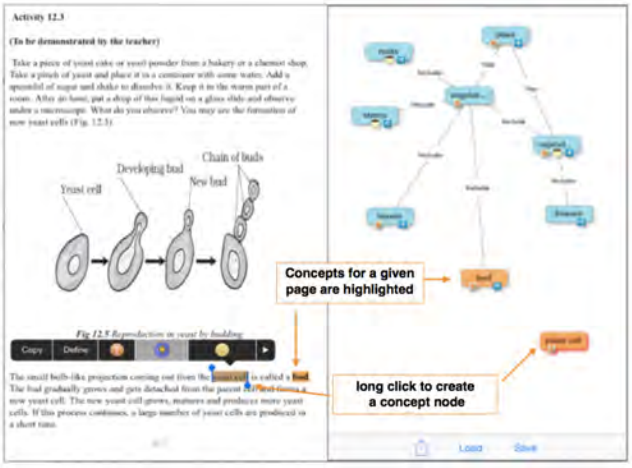


Figure 2. Interactive concept mapping system interface.

3. Results So Far

My previous study showed that types of interactions with the concept map have an effect on student learning gains [8]. However, limitations of the CmapTools and the complexity of the expert skeleton map reduced the effect of adaptive scaffolding. To solve these problems, I have implemented the proposed iPad-based concept mapping system and I'm currently running two studies to explore how different designs of expert skeleton maps and hyperlinking navigation effect learning out comes.

To test the effect of different types of scaffolding, I'm running an online study using Amazon Mechanical Turk, a human intelligent task market in which anyone can post tasks to be completed and specify prices paid for completing them. The literature indicates that Amazon Mechanical Turk could be a promising approach to get inexpensive, yet high quality data for research in psychology and social sciences [9]. However, few research has examined the

quality of Mechanical Turk data in educational studies. Thus, I plan to explore how Mechanical Turk can be used as a cost effective way to get high quality data for educational studies. In this study, participants use an online iPad simulator running the concept mapping application to construct a concept map while they learn a chapter of a high school science textbook. First, students are given a 2-minute pretest to assess prior knowledge on water pollution. Next, students are given a 3-minute training about what concept maps are and how to use the application to construct one. After the tutorial and practice, students are given a randomly modified expert skeleton map and are given 20 minutes to construct or complete the map based on the template. Finally, a posttest is given. Instead of tailoring the scaffolding specifically to student prior knowledge, I'm randomly selecting the size and concept nodes that appear in the template, in order to generate more variations of the expert skeleton map. Learning outcomes based on these different designs of expert skeleton maps could help us understand how the expert skeleton map should be designed to better facilitate learning.

Furthermore, I plan to examine how concept map-based navigation facilitates concept map construction and how it helps students to form personalized navigation patterns. I am currently working with a high school teacher to conduct a study in one of her classes, which has been using concept maps as a class activity. The study will last 20 minutes per day for 5 days and it will be a substitute for a paper-and-pencil based concept mapping activity. Students will construct the concept maps while they learn about the current textbook chapter. Students will be randomly assigned into two conditions: The hyperlinking condition, where nodes in the concept maps are hyperlinked with the textbook, and the non-hyperlinking condition. Pre and post tests will be given before and after the study. To investigate the effect of hyperlinking, I will compare the learning gains between condition. Furthermore, I plan to use data mining techniques to extract patterns within student navigation activities. For example, if a student is navigating by clicking back and forwards on two linked concept nodes, it might indicate that the student is using the textbook content to compare the concepts. If a student is navigating by clicking on a series of connected nodes, it might indicate that the student is comparing multiple concepts to understand some knowledge structure in a higher level.

4. Advice Sought

For this doctoral consortium, advice is sought regarding two major concerns. First, how should I validate the Amazon Mechanical Turk study results? I'm currently using Amazon Mechanical Turk platform for the expert skeleton map study. As I'm randomly varying the size and the concept nodes which appear in the template, I need a large number of participants to form overlaps between the student prior knowledge and the given expert skeleton map. Amazon Mechanical Turk would be a cost-efficient approach to get large amount data. However, due to the large variations in the participant population, the results from the study might not truly reveal the effect of expert skeleton map scaffolding on high school students. How could I make use of the Mechanical Turk study data to design concept mapping scaffolding to better facilitate learning?

Second, what data mining techniques can be used to analyze the hyperlinking study data? I'm interested in discovering what student behavior patterns correlate to learning outcomes and what interactions are tedious and counterproductive, and can be potentially be supported or replaced by computer technologies.

Problems discussed above are major challenges I encounter to analyze the data from the studies. Advice on these two problems will be very helpful to my work of designing personalized expert skeleton maps to facilitate concept map construction and providing hyperlinking navigation to reinforce student learning.

Acknowledgments

This research was funded by NSF CISE-IIS-1451431 EAGER: Towards Knowledge Curation and Community Building within a Postdigital Textbook.

5. REFERENCES

[1] Novak, Joseph D. "Concept mapping: A useful tool for science education."*Journal of research in science teaching* 27.10 (1990): 937-949.

[2] Azevedo, Roger, and Allyson F. Hadwin. "Scaffolding self-regulated learning and metacognition–Implications for the design of computer-based scaffolds." *Instructional Science* 33.5 (2005): 367-379.

[3] Chang, Kuo-En, Yao-Ting Sung, and S. F. Chen. "Learning through computer-based concept mapping with scaffolding aid." *Journal of Computer Assisted Learning* 17.1 (2001): 21-33.

[4] Chularut, Pasana, and Teresa K. DeBacker. "The influence of concept mapping on achievement, self-regulation, and self-efficacy in students of English as a second language." *Contemporary Educational Psychology* 29.3 (2004): 248-263.

[5] Novak, Joseph D., and Alberto J. Cañas. "The theory underlying concept maps and how to construct and use them." (2008).

[6] Cañas, Alberto J., et al. "CmapTools: A knowledge modeling and sharing environment." *Concept maps: Theory, methodology, technology. Proceedings of the first international conference on concept mapping.* Vol. 1. 2004.

[7] Wang, Shang, et al. "Personalized Expert Skeleton Scaffolding in Concept Map Construction." *Artificial Intelligence in Education*. Springer International Publishing, 2015.

[8] Burton, Richard R., and John Seely Brown. "An investigation of computer coaching for informal learning activities." *International Journal of Man-Machine Studies* 11.1 (1979): 5-24.

[9] Buhrmester, Michael, Tracy Kwang, and Samuel D. Gosling. "Amazon's Mechanical Turk a new source of inexpensive, yet high-quality, data?."*Perspectives on psychological science* 6.1 (2011): 3-5.

[10] Kinchin, Ian M. "If concept mapping is so helpful to learning biology, why aren't we all doing it?." *International Journal of Science Education* 23.12 (2001): 1257-1269.

[11] McClellan, James H., et al. "CNT: concept-map based navigation and discovery in a repository of learning content." *Frontiers in Education, 2004. FIE 2004. 34th Annual*. IEEE, 2004

Industry Track - Short Papers

Analysing and Refining Pilot Training

Bruno Emond
National Research Council Canada
1200 Montreal Road, Ottawa,
ON, Canada. K1A 0R6
1-613-993-0154
bruno.emond@nrc-cnrc.gc.ca

Scott Buffett
National Research Council Canada
46 Dineen Drive, Fredericton,
NB, Canada. E3B 9W4
1-506-444-0386
scott.buffett@nrc-cnrc.gc.ca

Cyril Goutte
National Research Council Canada
1200 Montreal Road, Ottawa,
ON, Canada. K1A 0R6
1-613-993-0805
cyril.goutte@nrc-cnrc.gc.ca

Ruibiao Jaff Guo
CAE Defense & Security
1135 Innovation Dr, Kanata,
ON, Canada. K2K 3G7
1-613-247-0342
jaff.guo@cae.com

ABSTRACT
Competency based training has become a major thrust in the development of instruction in both civilian and military pilot training. This paper reports on a joint effort by CAE and the National Research Council to identify data analytics methods relevant for the analysis, and refinements of competency based pilot training. In particular, these methods aim to identify correlations between 1) student actions and behaviours while engaging in training, and 2) students' success and incremental progression in the corresponding competencies being acquired. The paper presents some of our main results in applying sequence mining and additive factor modelling to small sets of pilot training data.

Keywords
Aviation pilots, competency-based training, sequence mining, additive factor models.

1. INTRODUCTION
Over the years, CAE has developed many research collaborations with universities and government research laboratories. The current paper presents some results from a project between CAE[1], the Advanced Technologies for Learning in Authentic Settings (ATLAS) research team from McGill University, and the Learning and Performance System Support program at the National Research Council Canada. The research efforts were focused on the identification of education data mining methods with practical outcomes for the improvement of pilot training. The main objective is to be able to analyse performance, and use competency models in order to refine simulation scenarios and CBT courseware. The contributions to the project represent different perspectives from sequence mining (descriptive method), to logistic regression models (predictive method). The objective was to explore the data from different points of view.

The following section presents an overview of the main trends in pilot training including competency, evidence, and scenario-based training. The next section briefly presents the data set that was used for all the analysis, and the remaining two sections presents the main results of applying sequence mining and additive factor modeling to this data.

[1] http://www.cae.com/about-cae/corporate-information/faq/

2. TRENDS IN PILOT TRAINING
To address the challenges of pilot training in the early 2000s, civil aviation stakeholders like the Civil Aviation Safety Alert (CASA), the International Civil Aviation Organization (ICAO), and concurrently the United States Air Force (USAF) have been promoting competency and evidence based training as a training model [1]–[3]. This position was in reaction to hours-based training where the number of flight hours or sorties done by a pilot determined flight or mission readiness. With the increase of flight operation complexities, it became obvious that achievement of a certain performance level on a task would be a better indication of a pilot competency, than the number of hours of practice, even though flight hours could be an indirect measure of a competency level.

There are many views about what a competency is. The International Civil Aviation Organization defines a competency as "a combination of skills, knowledge and attitudes required to perform a task to the prescribed standard" [4]. The USAF has developed an elaborate competency framework [5]. The Mission Essential Competencies (MEC) framework is intended to blend training task lists, and mission essential task lists. The MECs incorporate a wide range of pilot competencies, beyond the operational requirements, to include teams and inter-team competencies [3]. The Federal Aviation Administration (FAA) also recognizes that pilot competencies need to be defined at a higher-level than simply the low-level operations of an aircraft, especially with the increased level of automation because automated systems are not adapted to unforeseen situations [6]. Competency frameworks are usually the result of an analysis performed by subject matter experts who identify key competencies based on standards of performance and means to measure them.

Another important trend in pilot training is evidence-based training. The ICAO defines evidence-based training as "Training and assessment based on operational data that is characterized by developing and assessing the overall capability of a trainee across a range of core competencies rather than by measuring the performance in individual events or manoeuvres" [1]. The essential element evidence-based training introduces to competency based-training is the reference to operational data as a means to identify key competencies, in addition to the analysis

performed by subject matter experts. Evidence-based training applies the principles of competency-based training for safe, effective and efficient airline operations, while addressing safety threats. The term evidence refers to the fact that safety threats are identified from actual flight monitoring data, such as those provided by the Flight Operational Quality Assurance (FOQA) program, Aviation Safety Action Program (ASAP) data for business aviation [7], as well as Automatic Dependent Surveillance-Broadcast (ADS-B) data.

A literature review also revealed that a combination of competency, evidence, and scenario-based training approaches can form the basis for the next generation of pilot training system. The combination requires links between the development of simulated scenario events and performance measures, both driven by training objectives [8]**.** This combination is well integrated in the specification of evidence-based training as defined by the ICAO [1], and the focus on scenarios and simulations provides the foundation of a strong learner centred approach.

Simulation scenarios are central to evidence-based training as the main instructional content a trainee pilot interacts with, for evaluation and learning. The approach is consistent with the principles of situated learning theory, which argues that learning best takes place in the context in which it is going to be used. Scenario-based training is mostly suitable for procedure-oriented tasks requiring decision-making and critical thinking in complex situations, and is learner centered as the scenario provides a unique opportunity for the trainee to perform and acquire competencies based on his/her competency level.

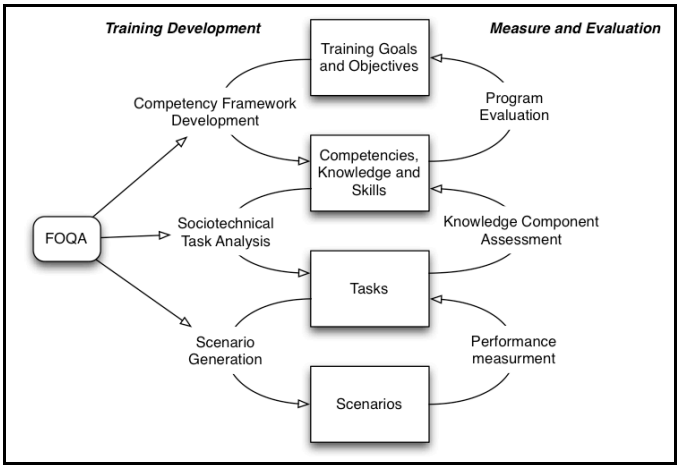


Figure 1. Competency, evidence and scenario-based training systems

Figure 1, inspired from [8]**,** tries to capture the relationships between competency-based training, evidence-based training as flight data monitoring programs feed in information for training development at all levels, and scenario-based training which constitutes an essential element for providing learner centered experiences. In addition to the closed workflow between A) training goals and objectives; B) competencies, knowledge, and skills; C) tasks; and D) scenarios, Figure 1 distinguishes on the left hand side training development including: the specification of competency frameworks, sociotechnical task analysis, and scenario generation. The right hand side of the figure presents key elements related to the measure and evaluation including: performance measurement, knowledge component assessment, and program evaluation.

The remaining sections of the paper fall essentially within the right hand side of Figure 1 under "Knowledge Component Assessment". The courseware delivery software gathered the student learning performance data during the learning process, including the sequences of activities selected by the students, timestamps, and question answers.

3. DATA DESCRIPTION

The data consists of two sets of web training sessions engaging students on scenarios requiring information gathering, review and assessment of new flight procedures with demands on both knowledge and skill acquisition related to taking off and landing operations. The two data sets correspond to two separate groups of students, and had respectively eight and six students in them. Table 1 presents the frequency distribution of events either as being assessments or information-gathering events for each student in the two groups. The counts in Table 1 refer to the sum of single events. For example, student 1 in Group 1 was assessed 46 times and gathered information 503 times. Essentially, information-gathering events refer to pages containing texts or videos, and assessment events refer to pages where an evaluation of knowledge or skills is performed. Overall the student pilots in the first group had a ratio of about 9% of assessment for information gathering events, while the pilot students in the second group had a ratio of about 13%. The number of assessments includes repeated trials on assessment items. Given that the following sections focus on specific subsets of observations (ex. frequent sequences, or first attempt assessments only), Table 1 provides a high-level view and context for these learning events analysis.

Table 1. Distribution of assessment and information events for each student in the two groups.

Students	Assessment	Information	Total
Group 1			
1	46	503	549
2	45	497	542
3	51	514	565
4	42	495	537
5	52	477	529
6	49	512	561
7	47	547	594
8	57	478	535
Group 1 Total	389	4023	4412
Group 2			
a	42	305	347
b	55	323	378
c	37	259	296
d	34	280	314
e	41	311	352
f	37	284	321
Group 2 Total	246	1762	2008
Grand Total	635	5785	6420

4. SEQUENCE MINING

The objective of the application of sequence mining techniques to the learner dataset was to test the hypothesis that students who acted similarly in training would also perform similarly in the assessments. Results indicate that a significant relationship between students' behavioural patterns during training and performance on test problems exists.

For the analysis in this section, we utilized a data-driven approach to classify student activity and behaviour patterns in the web training courseware, with the purpose of identifying dependencies between the way students interact with the training material, and how the students perform on subsequent assessment-based tests and exercises. At a high level, the working hypothesis for this part of the study is thus that students who behave similarly (i.e. by exhibiting similar patterns of navigation activity when interacting with the courseware) will perform similarly in the assessments.

To test this hypothesis, we classified the students into two groups, using three different criteria: 1) those who scored above the median score on the assessments versus those who scored below the median, 2) those who scored above average on assessments versus those who scored below, and 3) classification according to response similarity. For this final classification scheme, we considered similarities in student success on a question-by-question basis. A distance function was introduced, with the distance between two students defined as the number of assessment questions for which one student gave the correct response and the other gave an incorrect response. K-means clustering was then used to divide the students into two groups in which in-class distances were minimized. Thus two students in the same class were likely to have scored the same (correct or incorrect) more often than two students in different classes. This particular analysis thus more closely strives to validate the working hypothesis that students who behave similarly will perform similarly in the assessments. So, rather than only judging similarity between two students only in terms of total score, we also took a view of how they scored in relation to each other in terms of the number of assessments in which both responded correctly or both responded incorrectly.

For each classification scheme above, the hypothesis is that students classified in the same group (i.e. those whose score similarly in assessments in terms of total score or response similarity) should have exhibited more similarities in how they interacted with the courseware during the learning phase. To test this, we utilized sequential pattern mining (using the SPAM [9] algorithm) to mine sequences of behaviour that were discriminative of each group (i.e. sequences of pages visited that were found to be highly frequent in one group and highly infrequent in the other), and then used leave-one-out cross-validation to test our ability to correctly classify each student based on the existence of these mined behavioural sequences.

Figure 2 shows the accuracy of our classifier for each classification scheme. For example, the leftmost bar indicates that we were able to correctly classify whether a student scored above or below the median score in 93% of the cases (as well as above/below average in 100% of cases and according to response similarity in 86% of cases), solely through analysis of behaviour patterns exhibited by the students when navigating through the courseware. The p-value for each statistic indicates the probability of achieving these results (or better) purely by chance. This indicates that a significant relationship exists between students' behavioural patterns during training and performance on test problems.

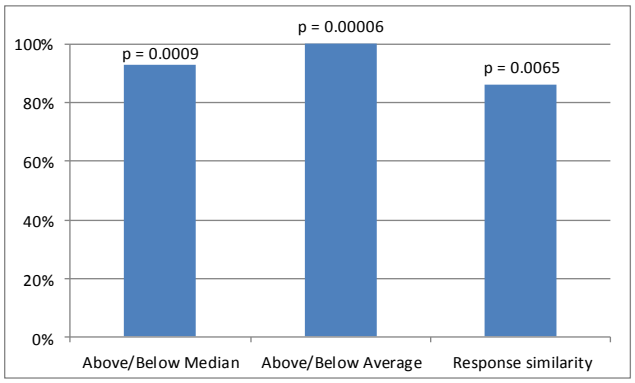


Figure 2. Results of sequence classification on students

To further examine the relationship between behaviour and results, we took a closer examination of the similarities between students when classified as either above or below average score, the scheme that was most successful in the test above. Here we generated the set of frequent behaviour patterns exhibited by each student, and then computed the Jaccard similarity of each pair by quantifying the degree of overlap in the set of frequent patterns for each student, where the Jaccard similarity of two sets A and B is equal to the size of the intersection of A and B, divided by the size of the union. Table 2 summarizes these results by showing, for each student, the average similarity to students who placed above and below the average. On average, students achieving a lower than average score had more similar behaviour to other students who achieved a lower than average score, and vice-versa. In fact, in all cases but one, each student behaved more similarly on average to students in its own group.

Table 2. Average similarity for each student to students with below/above average score

Below Average Students			Above Average Students		
Student	Similarity with below average students	Similarity with above average students	Student	Similarity with below average students	Similarity with above average students
1	0.125	0.080	3	0.059	0.071
2	0.078	0.068	4	0.100	0.075
5	0.047	0.033	a	0.051	0.068
6	0.070	0.061	b	0.063	0.112
7	0.032	0.026	c	0.024	0.042
8	0.127	0.075	d	0.063	0.133
			e	0.040	0.072
			f	0.059	0.142
Average	0.080	0.057		0.057	0.090

While there are wide-ranging behaviours that differentiate the two groups, Figures 3 and 4 point to two interesting behaviour patterns that were particularly prevalent in the initial dataset of 8 students. The first instance, in Figure 3, was highly frequent among the higher-achieving group, and quite infrequent among the lower-achieving group. This behaviour shows a lot of activity reviewing notes before completing a particular section and moving on. This could indicate that this note review had an impact on the success of the students. The second instance, in Figure 4, was highly frequent among the lower-achieving group, and quite infrequent among the higher-achieving group. This behaviour shows a lot of activity around calculations regarding take-off. This could provide

a clue into where the less successful students are going wrong, and thus where improvements to the courseware may be made.

```
1.  Review_Introduction_1, Review_Introduction_2,
2.  Full_Review_Notes_Mission_Planning_1,
3.  Full_Review_Notes_Landing_Limits_and_Procedures_2,
4.  Full_Review_Notes_Landing_Crosswinds_3,
5.  Full_Review_Notes_Takeoff_Procedure_4,
6.  Full_Review_Notes_Takeoff_Conditions_5,
7.  Full_Review_Notes_Takeoff_Crosswinds_6,
8.  Full_Review_Notes_Landing_Calculations_7,
9.  Full_Review_Notes_Takeoff_Calculations_8,
10. Full_Review_Notes_ControlUnit_Invalid_9,
11. Full_Review_Notes_ControlUnit_Calculations_10,
12. Transition_To_Test-GUI_MAP,
13. Lesson_Conclusion_Pass
```

Figure 3. Example behaviour of the higher-performing group

```
1.  Select_Calculation-Takeoff_Crosswinds_1-
2.  Select_Calculation-Takeoff_Pitch_1-Takeoff_Pitch_2,
3.  GUI_MAP-Calculations_Introduction_1-
    Calculations_Introduction_2-
    Calculations_Introduction_3- Invalid_11-Invalid_12,
4.  Invalid_14-How_To_Use_Introduction_1-
    How_To_Use_Introduction_2,
```

Figure 4. Example behaviour of the lower-performing group

This result has a number of implications. First, it demonstrates a tangible correlation between how students choose to navigate the courseware and how well they perform on assessments. Second, it establishes clear evidence that opportunities exist to predict student achievement during the learning phase, when remedial action can be taken to improve comprehension. Finally, the ability to identify the key behaviours that have the highest impact on how a student will perform can facilitate strategic managerial decision making on how to direct the flow of student activity through the courseware.

5. ADDITIVE FACTOR MODELS

The Additive Factor Model (AFM) was chosen because it represents a common technique in educational data mining [12]. By using this data analysis technique, we were seeking estimations for parameters for student proficiencies, as well as items difficulty, and competencies easiness. AFM is a model for assessing the quality of an items-to-skills mapping, based on its ability to predict empirical observations of student results [10]. It may be seen as a generalization of Item Response Theory [11], where the response depends not only on item difficulty and student proficiency, but also on underlying knowledge components (KC) and the sequence in which they are met. In AFM, these knowledge components can be associated with competencies, skills, or declarative knowledge that are responsible for a student's performance. The mapping between an item (question, task, problem) and knowledge components is provided in the form of a binary Q-matrix $\mathbf{Q}=[q_{ik}]$, where $q_{ik}=1$ indicates that item i is associated to knowledge component k [13]. The probability that a student j will correctly answer an item i is modelled using a mixed-effect logistic regression

$$P(Y_{ij}=1|\alpha,\beta,\gamma)=\frac{1}{1+\exp\left(-(\alpha_j+\sum_k\beta_k q_{ik}+\sum_k\gamma_k q_{ik}t_{jk})\right)} \quad (1)$$

where α_j is the proficiency of student j (higher proficiency yields higher success rate), β_k is the easiness and γ_k the learning rate for knowledge component k (higher easiness yields higher success, higher learning rate means increased success on subsequent trials)[2]. The observed student sequence is summarized in the opportunity t_{jk}, i.e. the number of times student j has met knowledge component k. As learning progresses, increasing opportunity translates into higher probability of success in items associated with that KC.

Our learner dataset contains 38 items, taken by 14 students (in two sessions of eight and six) between zero and four times each, resulting in 533 transactions.[3] The course designers provided the Q-matrix mapping the 38 items to 14 knowledge components (Figure 5, where the items are the specific questions or problems that the students had to answer or solve, while the knowledge components are the underlying knowledge and skills accounting for the learner's performance on those questions or problems.

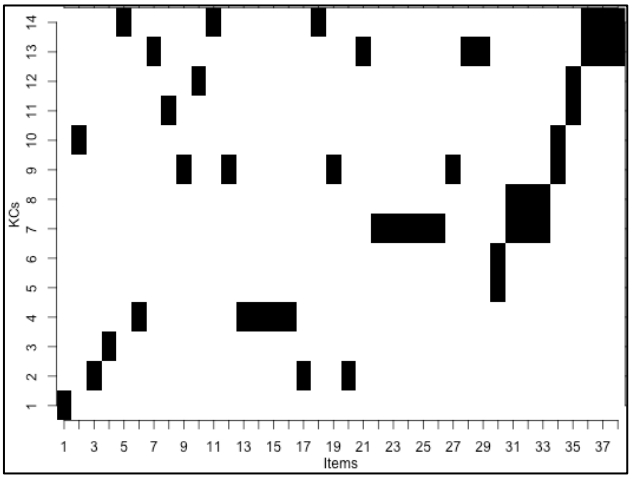


Figure 5: Q-matrix from courseware designer: 38 items x14 KCs.

Estimation of the AFM model parameters is done by maximizing the likelihood[4] on the transactions, with the constraint that learning rates are kept positive, and a slight regularization on the alpha parameters in order to keep them within the [-3; 3] range.

5.1 Student Proficiency

We analyse the proficiency of the two groups of students using the estimated alpha parameters. Figure 6 shows that the first group of students (1-8) has overall a lower proficiency than the second group (a-f). The two students with lower proficiency in the second group (b and c) have estimated proficiencies on par with the best two students from the first group (3 and 4). Student 5 clearly displays the lowest proficiency by far.

This is partly reflected in the observed success rates, which range from 58.5% for student 5, to 100% for student d. We learned *post analysis* that the second group had received an improved set of instructions. Although there was no difference between the first and second groups in expectations, motivation or engagement with the training material, the improved instructions have a clear

[2] Proficiency and easiness values are relative to the other values in the set, and should not be interpreted as actual success rates.

[3] Each transaction records one student's result on one item.

[4] We use a conjugate gradient algorithm. Any optimization method would work similarly as the log-likelihood is convex.

impact on the estimated proficiency for the second group. This validates the effectiveness of the change.

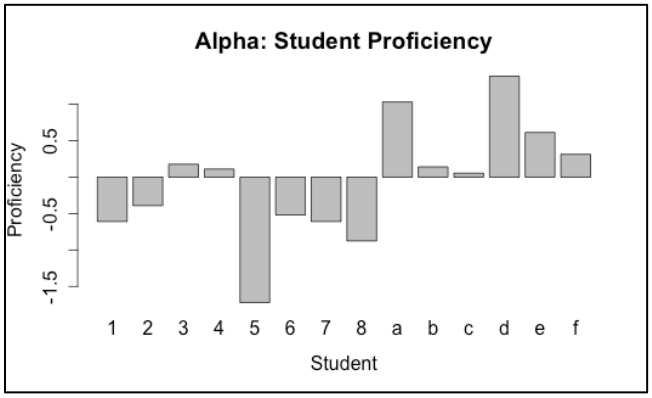


Figure 6: Student proficiency, estimated by AFM.

5.2 Competency Analysis

We analyse the competencies through the estimated beta and gamma parameters. Note that the actual parameter values are difficult to interpret separately, as various combinations of beta, gamma and opportunity may yield similar probabilities (Eq. 1). They do make sense in combination of the base "easiness" beta and learning rate gamma, to explain how the probability of success changes as the number of opportunity increases. As a consequence, rather than looking at actual parameter values, we relate them to the corresponding prediction ability. We analyse competencies by looking at the probability to fail on items associated by each knowledge component on the first three opportunities, for a hypothetical student with a proficiency parameter of zero. Figure 7 shows this for 11 knowledge components (The easiest KCs, 1, 4 and 11, get 0% for both predicted and observed error from the first attempts).

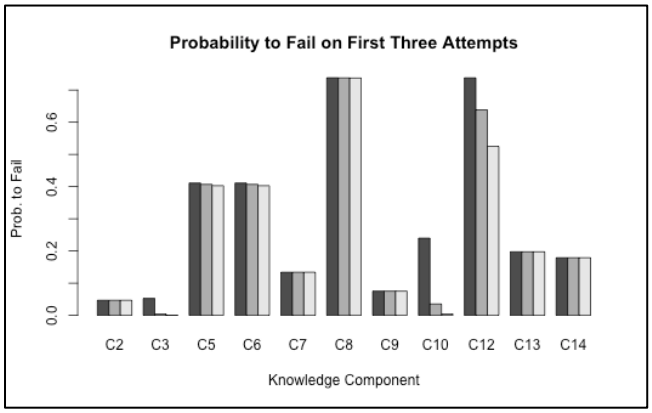


Figure 7: Probability of error for several knowledge components.

Note that due to the constraint that the learning rate is positive the probability to fail is always decreasing (Eq. 1). Learning is clearly apparent for several competencies (C3, C10 and C12), as shown by the clear drop in probability to fail as the KC is addressed. For C5 and C6, learning is much slower, and the error rate stays around 41%. However, this observation should be mitigated by the fact that these knowledge components are only associated with one item and always together (Figure 5). There is therefore very little data to estimate learning on these competencies, as most students took that item only once. When considered in combination in item #30, KCs C5 and C6 yield a predicted error on this item of 36%. In addition, this points to a possible refinement of the Q-matrix: these two knowledge components could be merged with no loss of modelling capacity.

Probability of failure seems consistently high for C8. However, Figure 5 shows that this knowledge component always appear together with C7 (which also appears alone). Due to the additive nature of the AFM model, the actual probability of success for items featuring C8 actually combine the easiness and learning rates for both C7 and C8, resulting in a probability of failure of 30.3%. Items involving both C7 and C8 are significantly harder than items involving C7 alone, and the AFM model adjusts for this fact by estimating a low easiness (high difficulty) for knowledge component C8.

The analysis of the AFM results therefore provides us with non-trivial insight into 1) the proficiency of the students taking the course, and 2) the difficulty and learning rates of the various competencies addressed in the course. It also suggests possible refinements of the competency framework produced by the course designer. Finally, despite the clear difference between the two groups of students, we have also observed that the estimates for the parameters related to competencies (β_k and γ_k) are consistent across the two groups.

6. CONCLUSION

To address the challenges of pilot training in the early 2000s, civil aviation stakeholders like CASA, ICAO, and concurrently the USAF have been promoting competency-based training as a training model. In addition to focusing on competencies rather than hours, the industry has also brought to bear actual flight monitoring data as a source to determine learning objectives. The essential element evidence-based training introduces to competency based-training is the reference to operational data as a means to identify key competencies, in addition to the analysis performed by subject matter experts. A literature review also revealed that a combination of competency, evidence, and scenario-based training approaches can form the basis for the next generation of pilot training system. The latter approach being consistent with the principles of situated learning theory, which argues that learning best takes place in the context in which it is going to be used. The paper focused essentially on the assessment of knowledge components using sequence mining and logistic regression for the purpose of understanding learning processes and improving learning scenarios. The data used for these analyses was collected in the context of pilot training using a scenario-based approach for reviewing basic landing and taking off flight operations.

The objective of the application of sequence mining techniques to the learner dataset was to test the hypothesis that students who acted similarly in training would also perform similarly in the assessments. Results indicate that a significant relationship between students' behavioural patterns during training and performance on test problems exists.

The Additive Factor Model, a model for assessing the quality of an items-to-skills mapping based on empirical observations of student results, was used to estimate student proficiency and knowledge components difficulty. Our analysis indicated a clear difference between students from two groups in the data. It also helped us identify competencies that are inherently easy, as well as hard competencies for which learning allows the probability of failure to quickly drop over subsequent attempts. It also suggests changes in the competency framework in which knowledge components could be merged with no loss of modelling capacity.

Together, the application of the descriptive method of sequence mining, and the predictive technique of additive factor models, provide results that may be used to evaluate and improve instructional design.

Some potential future directions for the project include: a) collecting more data, using the same approach for additional data sets, and comparing the result; b) developing alternative methods, and using the methods on same data sets to test and compare results; and c) conducting validation with instructional design experts in the relevant domain.

7. ACKNOWLEDGMENTS

The NRC project team would like to thank Dr. Susanne Lajoie (McGill University), who helped the NRC team to obtain its ethics certificate by providing the relevant documentation supporting McGill's ethics request to process CAE pilot learning data. The authors would also like to thank the following reviewers from CAE: Paula Mazzaferro, David Graham, and Graham Estey.

8. REFERENCES

[1] International Civil Aviation Organization, *Manual of Evidence-based Training*, First edit. Montreal, Canada: International Civil Aviation Organization, 2013.

[2] Civil Aviation Safety Authority, "Competency Based Training and Assessment in the Aviation Environment," 2009.

[3] C. M. Colegrove and G. M. Alliger, "Mission Essential Competencies : Defining Combat Mission Readiness in a Novel Way," in *RTO SAS Symposium on "Air Mission Training Through Distributed Simulation (MTDS) Achieving and Maintaining Readiness,"* 2002, vol. 323, p. 22.

[4] International Civil Aviation Organization, *Quality Assurance Manual for Flight Procedure Design. Flight Validation Pilot Training and Evaluation (Development of a Flight Validation Pilot Training Programme)*, First edit., vol. 6. Montreal, Canada: International Civil Aviation Organization, 2012.

[5] R. Chapman and C. Colegrove, "Transforming operational training in the Combat Air Forces.," *Mil. Psychol.*, vol. 25, no. 3, pp. 177–190, 2013.

[6] Air and Space Academy, "Dealing with Unforeseen Situations in Flight," Bruguières, France, 2013.

[7] M. Thurber, "The future of pilot training," *Aviation International News Online*, 2014. [Online]. Available: http://www.flightresearch.com/pdfs/AIN_Pg_20-28.pdf. [Accessed: 01-Feb-2015].

[8] J. MacMillan, E. B. Entin, R. Morley, and W. Bennett, "Measuring team performance in complex and dynamic military environments: The SPOTLITE method.," *Mil. Psychol.*, vol. 25, no. 3, pp. 266–279, 2013.

[9] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu, "Sequential pattern mining using a bitmap representation," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 429–435.

[10] H. Cen, "Generalized Learning Factors Analysis: Improving cognitive Models with Machine Learning," Carnegie Mellon University, 2009.

[11] R. D. Bock, "A Brief History of Item Theory Response," *Educ. Meas. Issues Pract.*, vol. 16, no. 4, pp. 21–33, 1997.

[12] A. Pena-Ayala, "Educational data mining: A survey and a data mining-based analysis of recent works," *Expert Syst. Appl.*, vol. 41, no. 4 PART 1, pp. 1432–1462, 2014.

[13] K. K. Tatsuoka, "Rule space: an approach for dealing with misconceptions based on item response theory," *J. Educ. Meas.*, vol. 20, no. 4, pp. 345–354, 1983.

A Scalable Learning Analytics Platform for Automated Writing Feedback

Jacqueline Feild
McGraw-Hill Education
Boston, MA
jacqueline.feild
@mheducation.com

Nicholas Lewkow
McGraw-Hill Education
Boston, MA
nicholas.lewkow
@mheducation.com

Neil Zimmerman
McGraw-Hill Education
Boston, MA
neil.zimmerman
@mheducation.com

David Boulanger
Athabasca University
Edmonton, CA
david.boulanger
@dbu.onmicrosoft.com

Jeremie Seanosky
Athabasca University
Edmonton, CA
jeremie
@rsdv.ca

ABSTRACT

In this paper, we describe a scalable learning analytics platform which runs generalized analytics models on educational data in parallel. As a proof of concept, we use this platform as a base for an end-to-end automated writing feedback system. The system allows students to view feedback on their writing in near real-time, edit their writing based on the feedback provided, and observe the progression of their performance over time. Providing students with detailed feedback is an important part of improving writing skills and an essential component towards solving Bloom's "two sigma" problem in education.

We evaluate our feedback system in two ways. First, we evaluate the effectiveness of the feedback for students with an ongoing pilot study with eight hundred students who are using the learning analytics platform in a college English course. In addition, we process an existing set of graded student essays and analyze the performance feedback. Results show a correlation between feedback values and human graded scores.

Keywords

Analytic Tools for Learners; Automated Essay Feedback; Scalable Analytics; Performance Feedback; Natural Language Processing

1. INTRODUCTION

Performance feedback is essential for self-regulated learning, which is an attribute of highly effective learners [3, 18]. Bloom has shown that providing formative feedback to students increases performance, compared to only providing final feedback [1]. This allows students to develop and implement actionable strategies for improving performance as they progress. Formative feedback is even more effective if it can be given in near real-time [7, 13].

In this paper we describe a scalable platform for learning analytics called OpenACRE (Analytics Collaborative Research Environment) which is currently in development to be released as open source. OpenACRE allows for ingestion of heterogeneous educational data from multiple source systems, long-term storage of raw data, running arbitrary models on the raw data using a parallel analytics engine, and short-term storage of resulting analytics for use by students, teachers, and researchers. As a proof of concept, we implement an end-to-end writing feedback system utilizing OpenACRE. Writing feedback is especially hard to provide in real-time and at scale as it is computationally expensive, making it well suited for the capabilities provided by OpenACRE.

There are several other existing writing feedback systems which provide various feedback to students, for example Revision Assistant, WriteToLearn, and Writing Pal [17, 14, 12]. While these systems provide useful information, they are either commercial black boxes which do not allow for modification, or are intelligent tutoring systems which provide writing instruction through customized modules. OpenACRE stands apart by providing the ability to develop and deploy new analytical models at scale, making it useful for researchers to test new feedback algorithms, predictive models, or reporting dashboards on a large number of students.

To evaluate our proof of concept system in a classroom setting, an efficacy study is currently underway to investigate the usefulness of the feedback to improve student performance. The study consists of eight hundred college students who are learning English at VNR VJIET in India. Additionally, we evaluate the feedback from 13,000 existing student essays and compare it to the human graded scores.

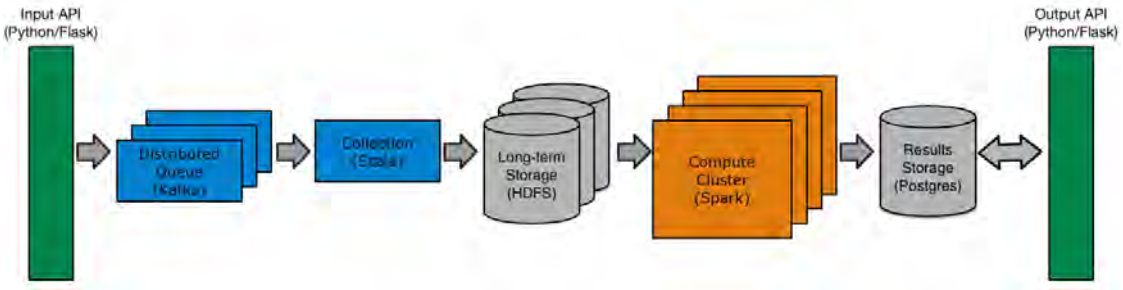


Figure 1: Architecture diagram of the learning analytics platform, corresponding to the middle box in Figure 2. Data is ingested by the input API and placed into a distributed queueing system which is implemented using Kafka. A collection service, implemented in Scala, pulls data from the queue and stores it in long-term storage, which is implemented using Hadoop Distributed File System (HDFS). The compute cluster runs models in parallel on the data in long-term storage and persists output views to the results store, implemented in PostgreSQL. Output views can then be accessed through the output API. Both the input and output APIs are RESTful and implemented in Python using Flask.

2. OPENACRE

The OpenACRE platform consists of an input and output API, long- and short-term databases, and a parallel computation cluster. A low-level diagram is shown in Figure 1. This platform is designed to handle the challenges of scalability, resiliency to data loss, and fault tolerance. Additionally, OpenACRE is built to be extensible for future models, without the need for drastic modification to the system as a whole. For example, models which perform machine learning algorithms, complex aggregations, and graph analysis could all be implemented to run on OpenACRE. These models could include traditional classroom statistics, score predictions, or personalized learning recommendations.

Learning event data is ingested into OpenACRE through the input API and persisted to the long-term data store. The input API for OpenACRE is implemented in a RESTful fashion using Python with the Flask package. RESTful APIs are used because they are stateless, easily extended for future functionality, and agnostic to programming language. The input API accepts event data from external sources and temporarily stores the events in a queueing system. We utilized open source Apache Kafka for our queueing system as it is distributed, durable, and supports APIs in several commonly used languages. Next, a collection service takes events from the queue and stores them in a long term data store. Here we use the open source Hadoop Distributed File System (HDFS) since it is distributed and fault tolerant. The collection service in OpenACRE is written in Scala, but any language supported by the Kafka and Hadoop APIs could also be used. The event data stored in HDFS is kept in its original "raw" form and is never altered. Storing unaltered event data allows for arbitrary computation and the implementation of future models without knowledge of those models beforehand.

Next, the computation engine runs analytical models by taking data from the long term store and performing transformations/aggregations to create new output views. These output views can be accessed by users through the output API. Open source Apache Spark was used for our computation engine as it allows for user-friendly parallel computation, horizontal scalability on commodity hardware, and contains a rich set of APIs ranging from simple map-reduce to machine learning algorithms. Additionally, Apache Spark currently implements APIs in Java, Python, and Scala.

Output views from a given model are written to the results store database which is implemented using PostgreSQL in OpenACRE. PostgreSQL was used as it is open source, has APIs in several languages, and provides a familiar SQL interface for queries. From the results store, output views are provided to external users through the output API. Similar to the input API, this is implemented as a RESTful API so it is stateless and can be easily accessed from the majority of modern languages. The output API can then accessed by other backend systems or user facing systems, such as dashboards. The combination of all the OpenACRE components listed above results in a learning analytics platform which can ingest arbitrary learning event data, apply parallel analytic models to the data, and provide the results of the analytics to external systems and dashboards in a generic fashion.

Figure 2: High-level diagram of the end-to-end writing feedback system. The learning management system and feedback visualizations are student-facing while the learning analytics platform stores writing data and computes feedback.

While any type of data format could be ingested into OpenACRE, we chose the standardized learning event format called Caliper, supported by IMS Global [4]. Caliper defines a set of standard learning events composed of actor-action-object triples. An example event is 'student-submits-quiz1'. While actor-action-object triples are also used in other standardized learning event formats like TinCan [16], Caliper has a significant benefit in that is uses JSON-LD, which is a schema-based JSON format. In addition to being schema-based, JSON-LD allows for easy mappings from JSON to domain-specific ontologies.

3. END-TO-END WRITING FEEDBACK SYSTEM

As a proof of concept, we built an end-to-end writing feedback system with OpenACRE at the core. Writing feedback is an excellent use case for OpenACRE as it is very computationally expensive, requiring approximately 12 seconds per essay for our feedback model. This large processing time results in almost 7 days of computation for a single assignment in a large MOOC of 50,000 students. Implementing our writing feedback model on OpenACRE allows that computation time to be cut to hours or minutes, depending on the size of the computation cluster.

The end-to-end proof of concept system includes the student facing system, which collects student writing data from their learning management system (LMS) and displays the automated feedback visualizations, and the backend system built on OpenACRE, which stores and analyzes the student data. Figure 2 shows a high-level view of this system, including both the student facing and backend systems.

The typical workflow for a student using this system includes:

1. Log in to writing course using an LMS
2. Start a writing assignment
3. Save the writing assignment
4. View visualizations of writing feedback
5. Edit writing assignment based on provided feedback
6. Save the writing assignment
7. Repeat steps 4-6 as needed
8. Submit assignment

This workflow provides feedback to students at regular intervals and gives students the opportunity to improve their writing before submitting their assignment. The ongoing pilot provides feedback in 24 hour increments due to cost constraints on the size of the computation cluster. Since the LMS which students are using is instrumented to directly collect writing data, there is no need to use an additional feedback system. This allows for an intuitive interaction between the student and their LMS, while collecting data for feedback at the same time. In our implementation, we utilized Moodle for our LMS as it is open source, familiar to both students and educators, and was easily instrumented to collect writing data as Caliper events and send those events to OpenACRE.

We designed a custom dashboard to display feedback visualizations to students and instructors. These include both a snapshot of overall feedback and the progression of feedback over time.

3.1 Feedback Competences

The feedback provided by our system is composed of seventeen writing competences which have been developed over the last several years [9]. These include traditional writing metrics such as spelling and grammatical accuracy as well as more advanced metrics that capture sentiment and writing flow. In the following sections, we describe several groups of writing metrics and define the competences we implement within them.

3.1.1 Traditional Metrics

Traditional writing metrics include competences that are often used by teachers to evaluate student writing. The competences implemented in our system from this category include vocabulary, spelling, grammatical accuracy, and lexical diversity. The vocabulary competence represents the amount of unique words in the student's text. As the student uses more unique words in their writing, the vocabulary competence increases. The spelling competence measures the percentage of incorrectly spelled words used. This competence increases as the percentage of misspelled words in the text decreases. Similar to spelling, the grammatical accuracy competence measures the percentage of grammatical errors in the text. This competence value increases as the percentage of grammatical errors decreases. Finally, the lexical diversity competence measures the percentage of unique words in the text. The value increases as students use more unique words relative to the size of the text.

3.1.2 Advanced Metrics

Advanced writing metrics highlight more subtle and complex characteristics of English writing. While not always explicitly listed in a writing rubric, these metrics are important for proficient English writing. The competences implemented in our system from this category include modifier complexity, noun phrase complexity, and tense agreement. The modifier complexity competence represents the amount of noun or verb modifiers which are used in the student's text. A high number of noun or verb modifiers indicates that the writing is more complex and expressive. The noun phrase complexity competence analyzes the number of noun phrases in the student's text. This metric attempts to measure the linguistic complexity for a piece of writing, as more noun phrases typically indicates richer sentences. Finally, the tense agreement competence measures the consistency of verb conjugations in the text. This competence value increases when verbs are conjugated consistently throughout a piece of writing.

3.1.3 Flow Metrics

Writing flow metrics measure how ideas are connected both within adjacent sentences and throughout entire pieces of text. The competences we implement in this category are

local cohesion, global cohesion, and connectivity. Local cohesion tracks the flow of ideas from sentence to sentence. Writing that contains adjacent sentences with similar nouns and verbs receives a higher local cohesion score. Similarly, global cohesion tracks the flow of ideas throughout an entire piece of writing, which is also measured by the similarity of nouns and verbs throughout the text. Connectivity measures the use of phrases that connect ideas to one another. Text with more coordinating conjunctions receives a higher connectivity score.

3.1.4 Descriptive Metrics

These writing metrics measure how descriptive a piece of writing is in several different ways. The competences we implement in this category are concreteness, imagery, familiarity, and conciseness. Concreteness measures the degree to which the text refers to tangible objects. Higher concreteness scores are obtained by using more words that refer to tangible objects. Imagery gives a measure of the amount of words within the text which evoke a mental image. Similarly, familiarity measures the amount of words in a text that are commonly used. The calculation of concreteness, imagery and familiarity are based on pre-defined scores in each category for commonly used words. These pre-defined scores were determined experimentally by asking human subjects to rate words in these three categories [5]. Conciseness measures the ratio of content words in the text. Writing that includes more nouns, verbs, adverbs and adjectives receives a higher conciseness score.

3.1.5 Sentiment Metrics

Sentiment metrics reflect the tone or feel of a piece of writing. These are computed using state-of-the-art techniques with the Stanford CoreNLP library [10, 15]. The required sentiment may vary based on the type of writing or subject matter. The competences we implement in this category include negative tone, neutral tone, and positive tone. The negative tone competence describes the degree to which the writing exhibits negative sentiment. Similarly, the neutral and positive tone competences describe the degree to which the writing exhibits neutral or positive sentiment. All three of these competences measure the amount of negative, neutral, or positive words in the writing.

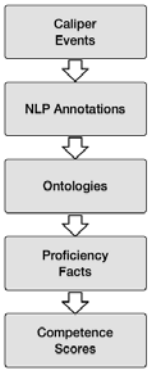


Figure 3: High-level diagram showing the flow of the openSCALE algorithm from caliper events to competence scores.

3.2 OpenSCALE

The analytics model implemented in this automatic writing feedback system is called OpenSCALE [2]. This model parses text with Stanford CoreNLP library [10], creates ontologies and facts from the annotated text, and aggregates the facts into competence scores for students.

A high-level view of the transformations which go from text to competence scores is described in Figure 3. First, the text is annotated using the Stanford CoreNLP library [10]. The annotations include tokenization of the text into words and sentences, part of speech tagging, syntactic parsing and sentiment analysis. These annotations are used to create an ontology of the relationships between words, sentences and paragraphs in the text, including both their structure and semantic meaning. For each piece of text, openSCALE creates one ontology using the open source Apache Jena library.

Next, each ontology is put through an inferencing layer, which looks for patterns in the ontology that show evidence of students having a particular skill/competence and creates proficiency facts. Each fact includes information about the degree of competence (weight) for a unique student-assignment attempt-time. Many facts are generated from a single ontology going through the inferencing layer. The inferencing layer in openSCALE is implemented using VIStology's BaseVISor framework [11]. BaseVISor works by passing a set of rules dictating how facts are generated for a given ontology. The ability for users to specify specific rules allows for great flexibility as different instructors could potentially dictate what is seen as evidence of different skills/competences. The current implementation of openSCALE uses a default set of rules which are used by BaseVISor.

Finally, the proficiency facts are aggregated to generate final scores for each competence. The main flow of the fact aggregations for student, competence, assignment attempt, and time is:

1. Sum the weights for all facts with the same student-competence-assignment attempt-time
2. For each fact F (student S - assignment attempt A - competence C - time T):
   (a) Find all facts at or before time T with the student S - competence C
   (b) Keep the facts of the newest attempt for each assignment
   (c) Sum the competence weights and update F

The final, aggregated facts are used to generate the competence progression view in the results store. The view displaying a snapshot of overall feedback is created by taking the latest aggregated facts for each competence.

4. PILOT RESEARCH STUDY

We are currently running a pilot research study to test the usefulness of the feedback system for increasing student writing performance. Eight hundred first year engineering students at VNR VJIET in India are using our system to complete up to twenty writing assignments.

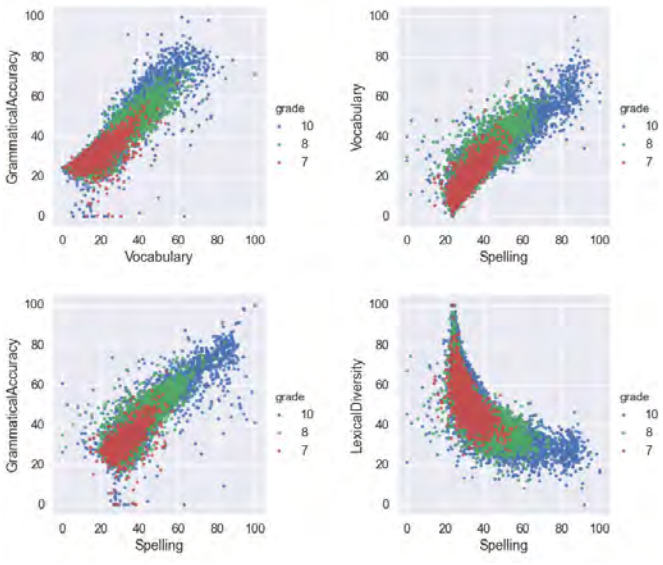


Figure 4: Scatter plots showing two competence scores plotted against each other all essays in the dataset. Data points are colored to distinguish essays from 7th, 8th, and 10th grades.

The current pilot study is an observational study and will use the method of propensity score analysis to determine the effectiveness of the feedback visualizations [6]. Students will also fill out surveys about the feedback they received and its usefulness.

5. ANALYSIS WITH EXAMPLE STUDENT ESSAYS

While the pilot is in progress, to additionally evaluate the usefulness of the writing feedback system, feedback was generated from a dataset containing about 13,000 anonymized student essays which have been graded by humans. The dataset was obtained from the Kaggle competition for automated essay scoring [8] and includes essays for students in 7th, 8th, and 10th grade. A total of eight different groups of essays are contained within the dataset, each with a different writing prompt and grading rubric. For our experiments, all essays were mixed together, grouped only by grade of the student, and all human grades have been normalized to range between 0-100.

First, we investigated correlations between competence types. Figure 4 shows competence vs competence scatter plots for grammatical accuracy, vocabulary, spelling, and lexical diversity. Data points are colored to distinguish between 7th, 8th, and 10th grade essays. Strong linear relationships can be seen for both plots containing grammatical accuracy in addition to vocabulary vs spelling. Additionally, an interesting relationship between lexical diversity and spelling can be seen in Figure 4. This plot shows that no students have high values in both lexical diversity and spelling simultaneously. To achieve high scores in the spelling competence, a longer essay is required with the majority of the words spelled correctly. In contrast, long essays tend to have lower lexical diversity competence values as more words are repeated in longer writings. The resulting balance of these two competences can be clearly seen in Figure 4.

Next, we plotted competence values against human graded scores. Figure 5 shows competence values for connectivity, grammatical accuracy, modifier complexity, and noun phrase complexity plotted against the graded score. Connectivity, grammatical accuracy, and noun phrase complexity all show the trend that increased competence values correlate to higher graded scores. The plot displaying modifier complexity shows the graded score initially increasing with competence value. There is a point which this trend stops and the average score stays constant, or even decreases, as the competence value increases. This data suggests that essays with a lot of complex modifier usage score the same or even lower than corresponding essays with moderate modifier usage. The above analysis gives us confidence in the usefulness of the competence feedback for improving performance.

6. CONCLUSIONS

Providing real-time feedback to students is an important component to solving Bloom's two sigma problem. In this paper we described a scalable learning analytics platform (OpenACRE) which is able to ingest educational data from multiple external systems and provide analytics on that data in near real-time. We demonstrated the usefulness of this platform with the implementation of a writing feedback system and are currently running a pilot research study to evaluate its effectiveness with eight hundred first-year engineering students at a university in India. We also showed that competence values correlated with human graded scores on a set of existing student essays. Development is currently underway to release OpenACRE as an open source project for other educational researchers.

7. ACKNOWLEDGMENTS

This paper is based on work supported by the McGraw-Hill Education Digital Platform Group (MHE DPG). Despite provided support, any opinions, findings, conclusions

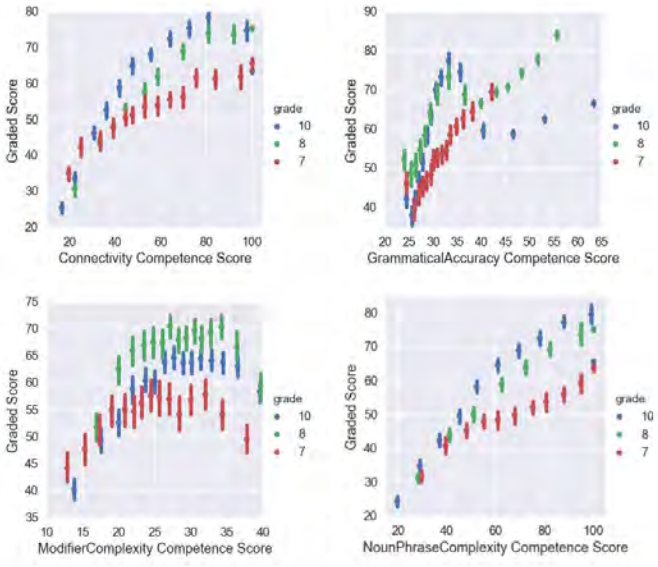


Figure 5: Average essay score as a function of several competence values. Error bars display standard deviation from the mean score. Data points are colored to distinguish essays from 7th, 8th, and 10th grades.

or recommendations expressed in this paper are those of the authors and do not necessarily reflect positions or policies of the company.

8. ADDITIONAL AUTHORS

Additional authors: Mark Riedesel (McGraw-Hill Education, Boston MA, email: `mark.riedesel@mheducation.com`), Alfred Essa (McGraw-Hill Education, Boston MA, email: `alfred.essa@mheducation.com`), Vive Kumar (Athabasca University, Edmonton CA, email: `vive@athabascau.ca`), Kinshuk (Athabasca University, Edmonton CA, email: `kinshuk@athabascau.ca`) and Sandhya Kode (IIIT Hyderabad, Hyderabad, India, email: `sandhya.kode@gmail.com`).

9. REFERENCES

[1] B. S. Bloom. The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational researcher*, pages 4–16, 1984.

[2] D. Boulanger et al. Scale: A competence analytics framework. In *State-of-the-Art and Future Directions of Smart Learning*, pages 19–30. Springer, 2016.

[3] D. L. Butler and P. H. Winne. Feedback and self-regulated learning: A theoretical synthesis. *Review of educational research*, 65(3):245–281, 1995.

[4] I. G. L. Consortium et al. Learning measurement for analytics whitepaper, 2013.

[5] K. J. Gilhooly and R. H. Logie. Age-of-acquisition, imagery, concreteness, familiarity, and ambiguity measures for 1,944 words. *Behavior Research Methods & Instrumentation*, 12(4):395–427, 1980.

[6] S. Guo and M. W. Fraser. Propensity score analysis. *Statistical methods and applications*, 12, 2015.

[7] J. Hattie and H. Timperley. The power of feedback. *Review of educational research*, 77(1):81–112, 2007.

[8] Kaggle. The hewlett foundation: Automated essay scoring. https://www.kaggle.com/c/asap-aes, 2012. Accessed: 2015-10-01.

[9] V. Kumar et al. Mobile computing and mixed-initiative support for writing competence. *Intelligent and Adaptive Learning Systems: Technology Enhanced Support for Learners and Teachers: Technology Enhanced Support for Learners and Teachers*, page 327, 2011.

[10] C. D. Manning et al. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[11] C. J. Matheus et al. Basevisor: A triples-based inference engine outfitted to process ruleml and r-entailment rules. In *Rules and Rule Markup Languages for the Semantic Web, Second International Conference on*, pages 67–74. IEEE, 2006.

[12] D. S. McNamara et al. The writing-pal: Natural language algorithms to support intelligent tutoring on writing strategies. *Applied natural language processing and content analysis: Identification, investigation, and resolution*, pages 298–311, 2012.

[13] D. J. Nicol and D. Macfarlane-Dick. Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in higher education*, 31(2):199–218, 2006.

[14] Pearson. The research behind writetolearn, 2007.

[15] R. Socher et al. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[16] R. Software. Tin can api. https://tincanapi.com, 2015.

[17] turnitin. Turnitin revision assistant, 2015.

[18] B. J. Zimmerman. Self-regulated learning and academic achievement: An overview. *Educational psychologist*, 25(1):3–17, 1990.

An Automated Test of Motor Skills for Job Selection and Feedback

Bhanu Pratap Singh
Aspiring Minds
bhanu.pratap@aspiringminds.com

Varun Aggarwal
Aspiring Minds
varun@aspiringminds.com

ABSTRACT

Motor skills are required in a large number of blue collar jobs today. However, no automated means exist to test and provide feedback on these skills. In this paper, we explore the use of touch-screen surfaces and tablet-apps to measure these skills. We design novel app-based gamified-tests to measure one's motor skills. We show this information to strongly predict the job performance of skilled workers in three different occupational roles. The results presented in this work make a strong case for using such automated, touch-screen based tests in job selection and to provide automatic feedback. To the best of the authors' knowledge, this is the first attempt at using touch-screen devices to scalably and reliably measure motor skills.

Keywords

Motor skills; Touch-screen devices; Tablets; Assessments; Blue collar jobs.

1. INTRODUCTION

There are many standardized automated tests of language, knowledge, cognitive skills and personality [8, 1, 2]. These tests, often taken on a computer, are good predictors of academic achievement and job performance in the knowledge economy. They have also enabled automated feedback and credentials for learners.

We are interested in automating assessments of motor skills required for vocational jobs such as tailoring, plumbing and carpentry. In the Occupational Information Network (O*NET) database of job descriptions [11], 350 out of 1,065 jobs need moderate to high motor skills. There has been tremendous interest worldwide among employers and professional organizations in training and efficiently identifying people that possess the skills for such hands-on occupations [3, 9]. There have been several validated, non-automated tests like the Purdue Pegboard test [13] and the O'Connor Tweezer Dexterity test [12]. However, no serious attempt has been made to develop and validate automated tests for this purpose. Automated assessments so far have exploited the power of PCs and laptops. We wish to make use of a touch interface, in the form of tablet devices, to test motor skills.

The ability to test motor skills automatically using touch interfaces would allow it to scale extremely well, given the high market penetration of inexpensive tablet devices in the last five years. This would enable people to measure their motor skills right from their homes and receive feedback toward self-improvement. There is substantial evidence that motor skills among adults can be improved [14] and that explicit motor skills feedback and instructions help do so [7, 5, 10]. Also, test takers can learn how suitable they are for a given job, get credentials for the skills they have acquired and apply for jobs that are the best match for their particular skill sets. Companies, for their part, can remotely administer these tests and can use the scores registered and the certificates offered to find a quality workforce, making the identification of suitable candidates easy, cheap, and scalable. This has the potential to make the blue-collar labor market considerably more efficient, similar to the effect automated testing has had on the white-collar labor market.

We apply the classical procedure used in developing skill assessments to develop tests which measure motor skills. We first identify the skills that are most useful to test. We then develop app-based tests that run on tablets and have the potential to measure these skills.[1] We use capacitive touch interfaces in this work, which are very popular these days. The app-based tests are designed in such a way that they *exercise* the motor skills of a person and are of varying degrees of difficulty. Candidates undergo testing through various movements of their fingers, hands and arms. We develop scores for each app based on the test taker's interaction with it. We then test whether these scores are predictive of job/task performance in three occupational roles: tailors, machinists/grinders and machine operators. If our test scores can indeed predict performance in job roles, they could be useful both to provide corporations with a way to filter/evaluate candidates for such jobs and to give feedback to job seekers and those interested in training for such specific fields.

We found that the app-based test scores can predict job performance across multiple parameters that are considered in

[1]We consider tablets instead of smartphones to assess wider movements of arms and shoulders.

evaluating the three job roles enumerated above. The correlation values range from 0.19 to 0.38. These are comparable to, and in cases outperform, those reported historically for manual motor skill tests in predicting job performance ($0.06 - 0.30$, Table 1). This provides strong support for the use of automated touch-screen tests for measuring motor skills for job selection and recruitment. The paper makes the following contributions:

- It is the first attempt to design a touch-screen based test of motor skills. We design a number of novel apps for this purpose.
- We show that there is firm supporting evidence for using app-based scores in the job selection/recruitment process for multiple jobs. This can yield tremendous scalability in the process of hiring blue-collar workers and providing them feedback.

This paper is organized as follows: §2 discusses the motor skills we measure; §3 discusses the design of our apps; §4 lays out the experiment objective and analyzes our results and finally, §5 concludes the paper.

2. MOTOR SKILLS TO MEASURE

We wished to identify motor skills that predict job performance for a range of jobs. We considered Fleishman's taxonomy of 52 human abilities [4] which includes skills such as verbal comprehension and selective attention. Ten of these, which are motor skills such as finger dexterity and arm steadiness, constitute the most widely recognized taxonomy of skills. These ten skills also figure prominently in the O*NET job and skill database.

It was found in [6] that four of these ten motor skills consistently predicted job performance based on empirical evidence. The four skills reported to correlate consistently with job performance are - finger dexterity, manual dexterity, wrist finger speed and multiple coordination (see Table 1). Detailed definitions of these skills can be obtained in [4]. In brief, finger dexterity refers to the accuracy in finger movements while manual dexterity refers to the speed of arm movements. Wrist finger speed refers to the speed of wrist and finger movements and multiple coordination refers to the proficiency in performing coordinated movements with two or more limbs.

A large number of manual tests have been used to measure these motor skills. In all these tests, a candidate is asked to perform a task and is rated on the time taken to complete it and the accuracy achieved, if applicable. For example, one test to measure manual dexterity requires a candidate to unscrew pegs from one board, turn them over and attach them to another board [6]. A test for finger dexterity requires a candidate to insert a rivet in a hole and secure it with a washer, where this process is repeated multiple times. These tests measuring motor skills correlate with job performance in the range of $0.06 - 0.30$ (Table 1).

We seek to develop automated assessments to measure these four skills, which could serve as an alternate to the manual tests described. Our intuition is that these skills involve movements of different joints: wrist/finger accuracy and speed - movements of finger and wrist joints; manual dexterity - movement of shoulder and elbow joints and multiple coordination - coordinated manual dexterity. We develop apps based on this intuition. We limited our work to the action of hands and no other limbs.

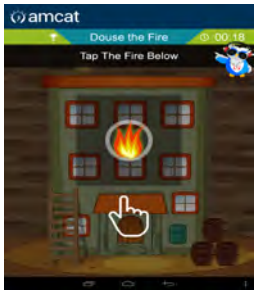
Skill	Correlations [min-max]	Weighted Mean Correlations
Finger Dexterity	$0.07 - 0.21$	0.19
Manual Dexterity	$0.08 - 0.24$	0.22
Wrist-Finger Speed	$0.14 - 0.30$	0.18
Multiple Coordination	$0.06 - 0.15$	0.14

Table 1: Skills and their minimum, maximum and weighted average correlation values with job performance [6].

3. DESIGN OF APPS

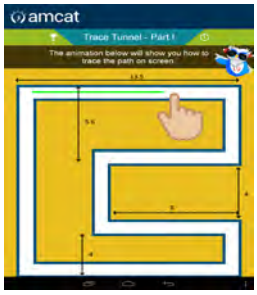
In this section, we describe the design of our touch screen apps to measure motor skills. We constructed each app to elicit specific hand and finger movements. We considered the simplicity and ease of comprehension of the apps as a key criterion. One should not be penalized for not understanding what has to be done, which could happen as a result of either cognitive or knowledge limitations. A set of instructions and a video/animation was shown before each app, to show how to perform the task. Each of these apps is described below:

1. **Douse the Fire (DOUSE)**: In this app, the candidate is shown 'fire' at random spots on a house shown on the screen (see Figure 1a). A candidate has to tap on the fire to douse it. As soon as the fire is doused at one spot, it appears at another spot on the house. In order to ensure that the fire occurs randomly, the distance between the two spots is probabilistically controlled using a uniform distribution between 0 and a number. The candidate has to douse as many fires within 30 seconds. We observed that the task requires elbow and shoulder movements and thus possibly measures manual dexterity.
2. **Trace a triangle-A (TRI_A)**: In this app, the candidate traces a path shown on the screen by dragging a finger over it. We initially considered having the candidate trace a line. However, we recognized that a candidate could not do this accurately because of the large surface area of the finger tip, restricting visual feedback of performing the activity incorrectly. We thus modified our exercise to contain two concentric equilateral triangles. The candidate was required to trace the path in between the triangles (see Figure 1b). The candidate was given feedback on the path traced by her through the use of colors. The path traced was green as long as it was confined to the designated area (space between the concentric triangles) and would turn red as soon as it went off the area. The width of the path is set to be more than the width of the fingertip (roughly 1 cm) to keep the task simple. The edge-lengths of the inner and outer triangles were 4.2 cm and 5.8 cm respectively. The candidate has



(a) Douse the Fire

(b) Trace Triangle-A



(c) Trace Path-A. (All dimensions are in cm)

(d) Roll Ball-B

(e) Resize

(f) Draw an arc-A

Figure 1: Snapshots of the apps

to trace as many triangles as possible in 30 seconds. As soon as one triangle was traced, another would appear. The app required moving one's hand quickly to trace the triangles and was designed to measure the speed element of manual dexterity. In principle, the task could be completed by finger movements, but we found that the default action made by the candidates which was comfortable to them involved shoulder and elbow movements.

3. **Trace a triangle-B (TRI_B)**: This app is similar to TRI_A with a difference that the width of the path was decreased. The width was kept a little lesser than the width of the finger tip. We hypothesized that the app required *careful* tracing and measured the accuracy element of manual dexterity.

4. **Trace a path-A and B (PATH_A and PATH_B)**: These apps are similar to the previous triangle apps. The difference is that candidates would trace over paths of much larger concentric polygons instead of a triangle, which shall require arm/hand movements (see Figure 1c). The polygons included rectangles, ellipses and those having zig-zag patterns. Figure 1c describes the dimensions of a sample path which was used. The path width shown in PATH_A is larger than those shown in PATH_B. The candidate has a maximum of two minutes to complete both the exercises and is required to trace as many polygons in the least possible time. These apps are designed to measure manual dexterity by tracing larger lengths and shapes, requiring different kinds of manual movements.

5. **Roll the ball-A (ROLL_A)**: In this app, a circle (symbolizing a hole) is positioned at the center of the screen and a ball is positioned at one of its corner. The ball rolls around on the screen on tilting the surface of the tablet. This is based on the tablet's accelerometer readings.[2] The candidate is required to guide the ball completely inside the circle. On the completion of one such exercise, the screen is refreshed with the ball placed at another point on the screen. The candidate has to complete four such exercises in the least possible time. The total time allotted is 40 seconds. The candidate moves the tablet with both her hands to guide the ball in the right direction. The test hence measures multiple coordination.

6. **Roll the ball-B (ROLL_B)**: This app is similar to ROLL_A. In this app, obstructions are placed in the path of the ball's movement (see Figure 1d). This is introduced to increase the degree of difficulty of the exercise. The time allotted to complete this exercise is 60 seconds.

7. **Fit a circle (FIT)**: We designed an app similar to the act of grabbing an object. The candidate is asked to perform a pinching action in a controlled environment. Two concentric circles were shown on the screen. The diameter of the inner concentric circle was fixed while that of the outer circle could be changed by the candidate. In order to change the diameter, the candidate had to place her thumb and her index finger on two points provided on its circumference and move them inwards or outwards without lifting them up. The diameter changed as the person dragged the two points.

[2]The accelerometer is calibrated at the beginning of the test by asking the candidate to place it on a flat table.

Skill Type	TT
Spot	Douse the Fire
Trace	Trace Triangle A and B
	Trace Path A and B
Multiple	Roll Ball - A and B
Grab/Pinch	Fit Circle
	Resize Circle
Rotate	Draw an Arc - A
	Draw an Arc - B

Table 2: List of tablet-based tests (TT).

The objective was to reduce the outer circle's circumference to match that of the inner circle. As soon as the two circles coincide, the screen is refreshed with two circles of different radii picked randomly. The candidate was required to perform this pinching action as many times as possible in 40 seconds. The app requires the rapid movements of fingers, say in grabbing many objects, one after the other and thus measures wrist-finger speed.

8. **Resize the circle (RESIZE)**: This is similar to the FIT app. The difference is that the outer circle now has to be shrunk and fit into a target ring as against placing it in a smaller concentric circle (see Figure 1e). On placing the outer circle within the target ring, the candidate is expected to lift her fingers from the screen, which then triggers the appearance of another target ring on the screen. The action is not considered until the fingers are lifted from the screen. This test measures the accuracy aspect, i.e. finger dexterity.

9. **Draw an arc-A (ARC_A)**: This app attempts to capture a candidate's wrist and finger rotation movement, as required, say, to screw or unscrew a nut and bolt. An arc is shown on the screen along with a pivot point (see Figure 1f). The candidate has to place her thumb on the pivot point and trace an arc shaped path with her index finger. On completing a trace, the screen is refreshed and a path with a different radius is presented. The candidate is required to trace six paths of varying radii in the least possible time. The arc paths are narrow (0.8 cm) requiring the candidate to be precise in her tracing. The entire task needs to be completed within 30 seconds. This test requires controlled and precise circular movements of the fingers. This test measures finger dexterity.

10. **Draw an arc-B (ARC_B)**: This app is similar to the ARC_A app but has wider arc paths. These arcs have 200% wider paths as compared to the arc paths presented in ARC_A. The candidate is required to trace as many arcs as possible in 30 seconds. This test requires rapid movement of wrists, say, in screwing a light bulb into a socket. This test measures wrist finger speed.

For each app, the candidate is instructed whether to place the tab on a table or hold it in her hands.

Skills measured	MST
Finger dexterity	O'Connor Tweezer Dexterity test [12]
Manual dexterity	GATB Manual Dexterity test [6]
Wrist-finger speed	Large Tapping test [6]
Multiple coordination	Purdue Pegboard test [13] We used the specific part of the test corresponding to coordination of both hands.

Table 3: List of non-automated manual motor skill tests (MST).

#	App	Score
1	Douse the Fire	Number of Correct douses
2	Trace Triangle - A	In-distance - Out-distance
3	Trace Triangle - B	In-distance
4	Trace Path - A	$\frac{\text{Time}}{\text{In-distance}-\text{Out-distance}}$
5	Trace Path - B	$\frac{\text{Time}}{\text{In-distance}}$
6	Roll Ball - A	$\frac{\text{Number of Rolls}}{\text{Time taken}}$
7	Roll Ball - B	$\frac{\text{Number of Rolls}}{\text{Time taken}}$
8	Fit Circle	$\frac{1}{\text{Number of fits}}$
9	Resize Circle	$\frac{1}{\text{Number of resizes}}$
10	Draw an Arc - A	$\frac{\text{Arcs}}{\text{Time taken}}$
11	Draw an Arc - B	In-distance

Table 4: Selected scores for each app. In-distance: Distance traced within path. Out-distance: Distance traced outside path.

4. EXPERIMENTS

We wish to answer whether the performance on tablet-based tasks can predict job performance. Specifically, we find out how our tablet-based tests and manual, non-automated motor skill tests compare in predicting job performance in industrial tasks like operating a lathe machine or tailoring clothes. This would act as a true indictor to suggest the practical use of the tablet-based tests in talent hiring. We note here that critical steps of non-automated motor skill tests like setting up the equipment, conducting the exercises and reporting scores are prone to human errors. Tablet-based tests have the distinct advantage of being devoid of such standardization issues. This advantage is likely to contribute towards its better predictive power.

4.1 Setup

The tests were administered to a workforce (referred to as *candidates* henceforth) belonging to three different occupations - tailors at a garment manufacturer, machinists and grinders at a machine-shop training company and machine operators at a skill training company. Each candidate was administered two sets of tests - tablet-based tests (TT henceforth) and non-automated, manual motor skill tests (MST henceforth). Four tests, as described in Table 3, were part of the MSTs. The standard set-up as described in [6] was followed in administering these tests. The eleven app-based tests described in §3 were part of the TTs.

TT and MST scores: In order to quantify a candidate's performance on our apps, we derived a single score for each

Job Performance Metrics	TT Scores				MST Scores				ATD Scores	
	Spot	Trace	Grab/Pinch	Rotate	MD	WFS	FD	MC	ATD	ATT
Tailors ($N = 74$)(Age range: $20 - 55$ years)										
Rate the tailor on the neatness of his/her completed work.	**0.22***	**0.37****	0.08	0.08	-0.09	-0.09	0.10	-0.10	NA	NA
Would you entrust him/her with a complicated task?	0.16	**0.33****	-0.10	-0.04	-0.08	**-0.33****	0.20*	-0.14	NA	NA
Rate how quickly s/he is able to complete her/his tasks.	**0.21***	**0.21***	-0.02	0.04	-0.13	**-0.20***	0.01	-0.13	NA	NA
Machinists and Grinders ($N = 68$)(Age range: $17 - 24$ years)										
Practical scores	**0.38****	**0.29****	**0.34****	0.07	0.07	-0.14	0.13	0.02	-0.06	NA
Electric Machine Shop score	**0.27****	0.11	**0.21***	-0.15	0.22*	**-0.29****	-0.03	0.10	0.12	NA
Machine Operators ($N = 78$)(Age range: $19 - 38$ years)										
Is s/he able to finish all the subtasks in a given operation?	0.15	**0.23****	0.00	-0.02	0.05	0.00	0.11	0.01	**0.20***	**0.27****
Rate how quickly s/he is able to complete the assigned operations.	0.17	**0.19***	0.04	-0.07	-0.14	**-0.19***	-0.01	-0.03	0.06	NA

* p<0.1; ** p<0.05; ATD : Attention to Detail scores; ATT : ATD + Best TT score;

FD - Finger Dexterity; WFS - Wrist-Finger Speed; MD - Manual Dexterity; MC - Manual Coordination.

Table 5: Correlations with job performance.

app (tabulated in Table 4). Further, the 11 tests were grouped into 5 skill types: *Spot* (DOUSE), *Trace* (TRI_A, TRI_B, PATH_A, PATH_B), *Multiple* (ROLL_A, ROLL_B), *Grab/Pinch* (FIT, RESIZE) and *Rotate* (ARC_A, ARC_B) (see Table 2). Each of these 5 skills was represented by a separate score. These scores were calculated by averaging the z-scores of apps contained in the skill[3]. For the four MSTs, scores were calculated as described in [6]. They generally measured the time taken to complete the task.

4.2 Data Set

The tests were administered to candidates belonging to three different occupations - 81 tailors, 74 machinists and grinders and 82 machine operators. The sample size was limited by the strength of the organizations. All three tests were administered by two event managers who had received a week's training on setting up the tests. Candidates performed the two tests (TTs and MSTs) with a gap of 5-6 hours. Each candidate's test was fully video-recorded. A review of these videos revealed that the standard process was not followed in 7.2% of the sample. These were discarded. The time recorded in nearly 3.7% samples for one or more of the MSTs was corrected. Post these changes, we finally had samples from 74 tailors, 68 machinists and grinders and 78 machine operators. We only considered the dominant hand in our analysis, except in the case for *multiple-coordination* which involves co-ordination between both hands. For machinists, grinders and machine operators, we also administered a multiple choice test of attention to detail (ATD)[4]. This was done to find what additional predictive power the TT scores added over the cognitive ability test scores to predict job performance.

Job performance scores: In the case of tailors and machine operators, a performance questionnaire (column 1 of Table 5) was developed on discussing with the candidates' managers. The managers were then asked to score the candidates on these metrics on a scale of 1 to 5. In the case of machinists and grinders, the training organization had documented scores from the candidates' lab-sessions. These scores were based on their performance on various job tasks given to them during their training. These ratings and scores formed the job performance data for our analysis.

4.3 Analysis and Observations

We compute the Pearson correlation coefficient (r) of all TT scores, MST scores and ATD scores (where available) with each metric contributing to job performance. The TT scores are fashioned to signal higher skill with higher magnitude whereas MST scores are fashioned to signal lower skill with higher magnitude. Hence, the correlation of job performance scores with TT scores is expected to be positive while the correlation with MST scores is expected to be negative. In our analysis, we observed the correlation between TT scores and MST scores to be in the range -0.27 to -0.34. This shows shared variance between the two scores. We noticed however that the scores of Multiple Coordination (one of the TTs) correlated positively with other MST scores. We hence do not include it in any further analysis. Additionally, by doing a regression, we found what incremental value the best correlating TT scores added over and above the ATD scores. These values and their respective significances are reported in Table 5.

First, and most importantly, we find that for every job per-

[3]Considering scores separately added no insight but increased complexity

[4]This is a criterion valid test used in hiring professionals in retail, sales, marketing etc. The 74 tailors had no formal education and hence could not take this test.

formance metric, at least one TT score shows a significant correlation (at $p \leq 0.1$) ranging from $0.19 - 0.38$ (mean: 0.27). This clearly establishes that TT scores are able to predict job performance and can be used for hiring/selection decisions by following standard practices. Second, MST scores show a significant correlation with four out of the seven performance metrics, where they range from -0.19 to -0.33 (mean:-0.25). We note here that the correlations between the four MSTs and job performance scores are in line with historically observed values (Table 1). ATD scores show a significant correlation in one case, where the *Trace* score adds significant incremental correlation (0.07) over and above it (column ATT, Table 5).

Among the app scores, there is maximum support for the *Trace* app which shows the highest correlation with job performance in five out of the seven metrics. In the remaining two metrics, the *Spot* app scores show the maximum correlation with job performance. While there is some support for the *Grab/Pinch* scores, there is hardly any support for the *Rotate* app scores. Among MST scores, the *Wrist-finger Speed* scores consistently correlate with job performance.

Discussion: We find that the TT scores are predictive of job performance in all cases in our study. The validity indices are comparable (and in cases bests) those observed for MST scores in the past (Table 1). The maximum support is for the *Trace* app. These are extremely encouraging results. This implies that the test may practically be used in making hiring decisions. The best way to do this would be to first perform a validity study with incumbents in a job in order to establish which TT apps distinguish on-job performance. These apps could then be used on new applicants and their scores be considered in the hiring process. While there is evidence for the *Trace* scores to be a universal predictor, the same may be established with further validity studies and meta-analysis. We envision that through such extended studies, a mapping could be formed between job roles and TT scores, akin to what has been established for MST scores. One would then know a priori which TT app and scores to use when hiring for a particular job role.

In four out of seven metrics, the MST and TT scores do equally well. One may observe that the MST scores did not do as well as the TT scores in three cases. This was surprising to us. A couple of reasons could explain this - first, the TT scores measure a larger variety of movements than the MST scores and some of these could potentially correlate better with job performance. For instance, there isn't any MST task similar to the structured tracing task in the TT. The other reason, as noted earlier, could be non-standardization and human errors in MST as compared to a controlled, completely standardized tablet-based test.

5. CONCLUSION AND FUTURE WORK

In this work, we explore the use of touch screen surfaces to measure motor skills. We show the scores of blue-collar workers on tasks performed on touch screen tablets to correlate with their respective job performances in the range of 0.19 to 0.38. These results make a strong case for using such automated, touch-screen based tests in job selection processes and in providing automated feedback. Such tests would make the process of identifying and credentialing skilled labor highly scalable and efficient, thereby benefiting both, individuals and corporations.

Our current work paves the way for substantial future work. The design of novel apps for motor skill measurement is a nascent area of research and could be further developed. By analyzing scores from such apps, we could create a map to suggest what scores are suitable for a given job role. Having such a map would help in automatically providing feedback to candidates on the skills they have. We could also perform the current tab tests for a number of other different job roles, which would help validate its design. Other devices and technologies such as smartphones[5] and resistive touchscreens could be experimented with, which could potentially make these tests more accessible, help do more accurate assessment and also grade new skills. For instance, a pressure detecting screen may help measure how soft the touch is, which might be relevant in nursing. We believe that the ideas introduced in this work can lead to substantial innovations in the blue-collar labor market.

6. ACKNOWLEDGEMENT

We thank Shashank Srikant for assistance and comments that greatly improved the manuscript.

7. REFERENCES

[1] J. Briel, K. OŠNeill, and J. Scheuneman. Gre technical manual. *Princeton, NJ: Educational Testing Service*, 1993.
[2] N. Claassen, M. De Beer, H. Hugo, and H. Meyer. Manual for the general scholastic aptitude test. *Pretoria: Human Sciences Research Council*, 1998.
[3] R. D. et al. The world at work: Jobs, pay, and skills for 3.5 billion people, 2012.
[4] I. Industrial/Organizational Solutions. Fleishman's taxonomy of human abilities, 2010.
[5] S. B. Issenberg, W. C. McGaghie, I. R. Hart, J. W. Mayer, J. M. Felner, E. R. Petrusa, R. A. Waugh, D. D. Brown, R. R. Safford, I. H. Gessner, et al. Simulation technology for health care professional skills training and assessment. *Jama*, 282(9):861–866, 1999.
[6] J. J. McHenry and S. R. Rose. Literature review: Validity and potential usefulness of psychomotor ability tests for personnel selection and classification. Technical report, DTIC Document, 1988.
[7] S. L. McPherson and K. E. French. Changes in cognitive strategies and motor skill in tennis. *Journal of Sport & Exercise Psychology*, 13(1), 1991.
[8] A. Minds. Amcat. `https://www.aspiringminds.com/`.
[9] A. Minds. Skills plumbers 2015 report, 2015. `http://www.aspiringminds.com/research-reports`.
[10] K. Mononen. *The effects of augmented feedback on motor skill learning in shooting: A feedback training intervention among inexperienced rifle shooters*. University of Jyväskylä, 2007.
[11] O. I. Network. O*net online, 1998. `https://www.onetonline.org`.
[12] J. O'Connor. Instructions for the o'connor tweezer dexterity test. *Indiana: Lafayette Instrument*, pages 1–5, 1998.
[13] J. Tiffin. *Purdue pegboard examiner manual*. Science Research Associates, 1968.
[14] L. G. Ungerleider. Functional mri evidence for adult motor cortex plasticity during motor skill learning. *Nature*, 377(6545):155–158, 1995.

[5]Most apps here can be used on a smartphone with some adjustment in the scale and aspect ratio of the apps and recalibration of scores. It may not effectively measure wider movements of the arms.

Industry Track - Posters

Studying Assignment Size and Student Performance Using Propensity Score Matching

Shirin Mojarad
McGraw-Hill Education
281 Summer Street,
Boston, MA USA
Shirin.mojarad@mheducation.com

ABSTRACT

Teachers and instructors assign students homework of varying lengths. There is considerable evidence that factors such as cognitive load play a role in student performance and learning, but there has not been sufficient study of how these phenomena play out in the specific case of the length of homework. In this paper, we study the impact of assignment size on student performance. This paper represents the first attempt we are aware of to study how long assignments should be, in real-world data, in order to maximize student performance and learning. However, natural assignments of different lengths often vary in other ways. We control for this limitation using propensity score matching (PSM), an approach that helps to control for variables affecting outcome besides the intervention of interest. As such, we can conduct our analysis on large-scale data naturalistically collected through a digital educational platform. We use PSM to study the effect of assignment size on student performance while controlling for assignment difficulty, discrimination and reliability. We find that shorter assignments result in higher performance. These results can be used as a guideline for instructors and instructional designers when designing course assignments.

Keywords

Propensity score matching, assignment size, classical item analysis, item difficulty, item discrimination, student performance, test reliability

1. INTRODUCTION

Graded assignments are used as an effective method to improve students' performance on final tests and improve learning [1]. Considering multiple shorter assignments as opposed to few, larger assignments is amongst the recommendations by USC for designing effective homework assignments [2]. This is because shorter assignments are less intimidating and help enhancing student motivation by minimizing the negative effects of a poor grade on student learning experience. In this study we investigate whether assignment size affects student performance. Since assignments of different sizes often vary in other ways, other assignment characteristics affecting the performance should be isolated to enable the study of assignment size effect on student performance.

Randomized control trials are considered the gold standard in conducting studies to investigate the effect of a particular intervention on a specific outcome [3]. However, their application is limited in educational settings as they can be conducted on a limited number of students. Results from the comparison of RCTs and OSs show that OSs can expand upon RCTs due to the use of large and diverse sample population [4]. Propensity score matching (PSM) is a common method in OSs to study the causal effect of an intervention on a particular outcome [4]. In this paper, we have used PSM to leverage the large amounts of data available through McGraw Hill Education digital platforms.

The goal of this paper is to study the impact of assignment size on student performance in isolation from other assignment characteristics including assignment difficulty, discrimination and reliability. This is the first effort of its kind in measuring an optimal assignment size to maximize student performance.

2. Materials and Methods

2.1 Data

We study these issues using data from assignments completed through McGraw-Hill Education's higher education platform, Connect. Connect is one of the most widely used digital platforms in higher education with over two million students and 25,000 instructors [5][6]. Connect allows instructors to design assignments in form of homework, practice, exams, or quizzes. Here, we refer to assignments as a set of items that either test student on knowledge and skills or allow students to practice what they have learnt on the course. Most Connect assignments are graded by the system automatically.

The dataset in this study is retrieved from all the courses created from the title Managerial Accounting 2nd Edition, by Robert Libby. We include all data for this title between September 2014 and January 2016. The original dataset included 362 classes, where 12,588 students responded on 3,072 items on 5,330 assignments, for a total of 1,031,298 student-item pairs. We have kept only assignments that have 10 or more student submissions. After applying this filter, there are 2,826 assignments left in the data. From the four of types of assignments in Connect, i.e. homework, practice, exam and quiz, we have focused on homework assignments. The reason is that in homework assignments, score is not as a strong motivator as exams and quizzes since homework assignments have a low weight in final score and are mainly aimed for development of self-study habits in students [7]. Hence, students are more motivated by learning to finish homework assignments. Therefore, size of a homework assignment will be an important factor in keeping students engaged throughout the assignment.

2.2 Exploratory Data Analysis

The dataset includes assignments' size, difficulty, discrimination, reliability, and average score where difficulty, discrimination and reliability are calculated using classical item analysis [8]. The assignment size in this dataset varies between 1 to 101 items.

Based on the rarity of very large assignments (and the likelihood that an assignment with over 100 items represents test practice or something different than briefer assignments), we have filtered down to assignments of size 16 or less. Filtering in this fashion still retains 98% of the assignments. We categorized assignments into short and long assignments by using a cut off for number of items within that assignment. Frequency of assignments drops for assignment sizes of larger than 5, which indicates most instructors prefer shorter assignments of size 5 or less. We have used this as a reference to decide a cut off value for number of items for short and long assignments. Following this definition, there were 1,787 short and 1,039 long assignments.

Figure 4 shows the mean score of different assignment sizes. As shown in this figure, the mean score of assignments drops as the assignment size increases.

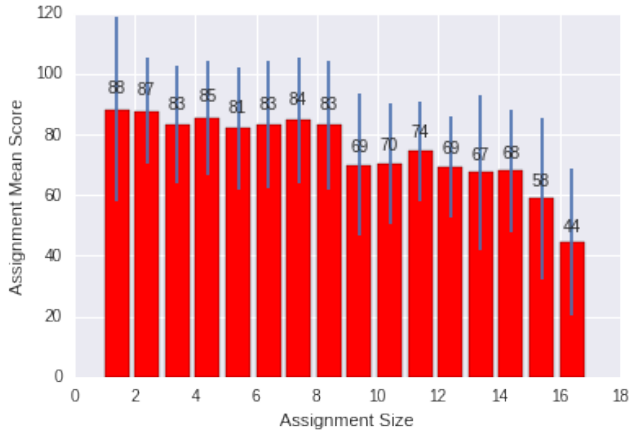


Figure 4. Assignment size versus assignment mean score

3. Results

Overall, students achieve an average 8.7 (on a scale of 0 to 100) higher score on short assignments than long assignments. When we control for difficulty, discrimination and reliability using PSM, students still achieve a 6.8 (on a scale of 0 to 100) higher average score on short assignments compared to long assignments.

The differences between the characteristics of short and long assignments matched using PSM are shown in Table 2. We have used Algina's d to compute the effect size of the difference of means between the two assignment groups [9].

As shown in this table, the effect size of difficulty, discrimination and reliability between two groups of assignments is negligible, indicating that these factors are no longer significant once we control for them using propensity score matching.

Table 2. P-value and the effect size of short versus long assignments, matched using PSM method

Attribute	Mean Difference	Effect Size (Algina's d)	P-value
Average Score	6.8	0.40	<0.001
Difficulty	0.00	0.00	0.99
Discrimination	0.00	0.01	0.53
Reliability	0.00	- 0.01	0.44

4. Conclusion

In this study, we investigated the effect of assignment size on student performance. Results of EDA show that student performance drops as the assignment size increases. The relation between assignment size and average score indicated that performance drops dramatically in assignments sizes of higher than 6. Hence, we used a cut off value of 6 to define short and long assignments. In order to investigate the statistical significance of this difference in two groups of assignments, in isolation from other factors affecting assignment performance, we used propensity score matching (PSM). The effect size and average performance difference of short versus long assignments is still significant when matching assignments with similar difficulty, discrimination and reliability. This indicates that longer assignments may increase cognitive load for students and negatively affect student performance and learning. These results can be used in form of recommendations to instructors when they are designing homework assignments on the Connect platform.

5. ACKNOWLEDGMENTS

This research paper is made possible through the help and support from Professor Ryan Baker, Dr. Lalitha Agnihotri and Alfred Essa, VP Analytics and R&D at McGraw-Hill Education.

This paper is based on work supported by the McGraw-Hill Education Digital Platform Group. Despite provided support, any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect positions or policies of the company.

6. REFERENCES

[1] Latif, E., Miles, S., (2011). The impact of assignments on academic performance. *Journal of Economic and Economic Education Research*, (Nov. 2011), 12 (3).

[2] Center for Excellence and Teaching, University of Southern California,http://cet.usc.edu/resources/teaching_learning/docs/teaching_nuggets_docs/4.2_Assignments_and_Homework.pdf, last accessed at March 2016.

[3] Silverman, S. L. (2009), From Randomized Controlled Trials to Observational Studies, *The American Journal of Medicine*, Volume 122, Issue 2, Pages 114–120.

[4] Rosenbaum, P. R., and Rubin, D. B., (1983). The Central role of the propensity score in observational studies for causal effects. *Biometrika*, 70 (1), 41-55.

[5] Feild, J., 2015. Improving student performance using nudges analytics. *Educational Data Mining*.

[6] Agnihotri, L., Aghababyan, A., Mojarad, S., Riedesel, M. and Essa, A., (2015). Mining Login Data For Actionable Student Insight. In Proc. 8th International Conference on Educational Data Mining.

[7] Sharma, Y. K., Fundamental Aspects of Educational Technology. Kanishka Publishers, Distributors New Delhi.

[8] Smith, Jeffrey K.. (1987). Review of *Introduction to Classical and Modern Test Theory*. *Journal of Educational Measurement* 24 (4). [National Council on Measurement in Education, Wiley]: 371–74.

[9] Algina, J., Keselman, H. J., & Penfield, R. D. (2005). An alternative to Cohen's standardized mean difference effect size: A robust parameter and confidence interval in the two independent groups case. *Psychological Methods*, 10, 317–328.

Toward Automated Support for Teacher-Facilitated Formative Feedback on Student Writing

Jennifer Sabourin
SAS Institute
100 SAS Campus Dr.
Cary, NC 27513
1.919.531.3313
Jennifer.Sabourin@sas.com

Lucy Kosturko
SAS Institute
100 SAS Campus Dr.
Cary, NC 27513
1.919.531.3430
Lucy.Kosturko@sas.com

Kristin Hoffmann
NC State University
Raleigh, NC 27695
1.919.515.7061
klhoffma@ncsu.edu

Scott McQuiggan
SAS Institute
100 SAS Campus Dr.
Cary, NC 27513
1.919.531.1119
Scott.McQuiggan@sas.com

ABSTRACT

Formative, content-level feedback on student writing has been shown to have positive impacts on both writing and learning outcomes. However, many teachers struggle to provide this type of feedback to large classrooms of students. This paper takes an initial step towards supporting teacher-facilitated feedback through the use of automated and user-directed topic discovery. 114 student essays were collected from a local underperforming middle school as part of a pilot study for Write Local, a digital repository and workspace for authentic problem-based learning activities. Predictive models were built and evaluated to explore the impact of different topic discovery approaches as well as correction of student spelling errors on model accuracy. The resulting models provide promising direction for scaffolding teachers in providing formative feedback on content-level features of students' problem-based writing.

Keywords

Problem-based writing, formative feedback, teacher-facilitated feedback, automated writing assessment, topic discovery

1.! INTRODUCTION

Problem-based writing tasks seek to elicit high-quality student writing by contextualizing the purpose of the task and providing an authentic audience [4]. These tasks also tend to extend across several days or learning periods offering more opportunities for formative assessment and feedback, which is expected to yield improved writing outcomes [1]. However, it is often difficult for teachers to focus on high-level features such as the focus, accuracy, and organization of student writing when working with a large classroom of students. Instead, teachers are more likely to focus on surface level features such as spelling, grammar, and mechanics. This is especially true in underperforming schools [2].

This work serves as an initial investigation into automated assessment of student writing in order to scaffold teachers in providing higher-level formative feedback. A pilot study was conducted as an initial step in the Write Local project. Write Local is intended to be a digital repository and workspace to facilitate both teachers and students in authentic problem-based writing activities. As part of a pilot study, 114 student writing samples were collected from students at an underperforming [3], local middle school as part of a multi-day problem-based learning activity. Student essays were manually coded for essay focus and accuracy. A variety of models for predicting these features were constructed and evaluated as an initial exploration for scaffolding teacher-facilitated feedback. In particular, this work sought to explore the role of automated and user-directed topic discovery in predicting content-level essay features. Additionally, we sought to investigate the importance of correction of student spelling mistakes prior to model construction. The results indicate that these initial models can serve as a starting point for supporting teachers in providing feedback on content-level features in problem-based writing and inform several directions for future work.

2.! PILOT STUDY

This investigation uses data collected during a pilot study of Write Local. Write Local seeks to employ crowdsourcing to ensure teachers and students have immediate access to a large repository of writing prompts that cover the entire spectrum of text types and audiences—persuasive, informative/explanatory and narrative. Local businesses, and in particular, those employing STEM-related positions, can post various letters of need as well as any supplemental documentation such as images or vocabulary lists. Teachers can then select a call from the repository and assign the project to their students. Students will use the integrated workspace to plan, research, document, draft, revise, present, and submit their response in one central space.

The entire sixth grade from a local, underperforming [3] middle school (54% free/reduced lunch) participated in this study as part of their regular social studies class. Of the 168 participants, 86 were male and 82 were female with a mean age of 11.5. Of the 168 participants, 114 completed all components of the procedure. For the remaining analyses only data from these 114 students is used.

For the study, students were divided by class into one of two conditions: experimental and control. On the first day of the study, students in the experimental condition viewed a 3-minute introduction video that contained problem context: a frozen yogurt company plans to open a new location and asked students to write a letter with their researched opinions about 1) which 5 toppings should be available on the topping bar and 2) where the new shop should be located. Students used authentic data and a map of the area to make their decisions. Students in the control condition were given a similar task without real-world contextualization. Students in both conditions were given two full 50-minute class periods to plan and write their letters.

Three researchers then transcribed and coded the essays with sufficient inter-rater reliability ($k = .89$). Essays were given a composite score for essay focus and accuracy. Using the final composite scores, students were divided into 3 evenly distributed categories (High, Medium, and Low) for both focus and accuracy. These groupings are intended to be presented to teachers to inform formative feedback for their students.

3.! TEXT ANALYSIS AND MODELING

The first step in building predictive models of student essay content classifications was to extract meaningful features from the student text. In total, the corpus for analysis included 114 student essays. The average length of the essays was 130.0 (SD = 91.4) words and 9.6 (SD = 7.6) sentences. The writing samples provided by the students were analyzed using SAS® Text Miner® and SAS® Enterprise Miner®.

For the purpose of this analysis we focused on the document topic analysis features of SAS Text Miner. The text topic procedure identifies terms that are strongly associated within the corpus. It also provides a strength of each topics' presence within the document. Topics can be automatically learned from the corpus or they can be provided or fine-tuned manually. Both approaches were used for this work. For automatic topic discovery, the limits were set at 25 multi-term topics. Manually-created topics were generated by highlighting terms in the text of the prompt and identifying whether each term applied to the problem context, the problem request, or the task instructions. In total 27 terms were identified; 8 context terms, 13 request terms, and 6 instruction terms. These terms were provided as user-created topics to the topic discovery procedure. In addition, up to 25 multi-term topics could be automatically generated; though because the engine tries to remove correlated topics, only 22 new topics were created. Of the 27 user-provided topics, only 17 occurred in the corpus of student data; 6 context terms, 9 request terms, and 2 instruction terms.

During essay transcription and coding, it was noted that there were a significant number of spelling errors present in the corpus. This may be due to the fact that essays were handwritten without the support of automated spell checking tools that many students are familiar with. In order to investigate the importance of correct spelling in modeling content-level features such as essay focus and accuracy, we chose to build models using different levels of spelling correction. Three different corpora of student essays were provided to the text topic discovery procedures: 1) the students' original texts, 2) an automatically spell-corrected version of the text, and 3) a manually spell-corrected version of the text.

For this exploration, we evaluated models across both topic discovery type (fully-automated and user-facilitated) and spelling correction type (manual, automated, and no correction). Additionally, we built separate models to predict both essay focus classification and essay accuracy classification. Finally, we used three modeling approaches for each corpus: logistic regression, decision tree, and neural network.

Each model was evaluated using 10-fold cross validation and predictive accuracies were compared against a baseline of most frequent class. This measure was 33.0% and 40.4% for essay focus and accuracy, respectively. The most common class for each evaluation type was Medium. With one exception, all models outperformed baseline with statistical significance at the 0.05 level (Table 1).

Overall, the models built using manual spelling correction and prompt-based topics outperformed other models in predicting essay focus and accuracy. This suggests that the prompt-based topics centered on the components of problem-based learning activities were beneficial in improving predictive accuracy. Unfortunately, this step requires manual annotation for each prompt. At present, this task, while manual, is not particularly labor intensive and can scale as we assess whether this benefit holds for future, unseen prompts. However, since the objective of Write Local is to scale with a large number of problem-based prompts, this may no longer be feasible. If we determine that this type of prompt annotation continues to be beneficial for predicting essay accuracy and focus we may investigate possible methods for automating or facilitating this task.

Table 1. Predictive accuracy for essay focus and accuracy using (a) discovered topics and (b) prompt-based topics

Discovered Topics

	Spelling Correction		
Model	Manual	Auto	None
Neural Net	F: 57.4 A: 55.0	F: 48.9 A: 51.9	F: 45.5 A: 52.6
Log. Reg.	F: 46.5 A: 56.1	F: 45.5 A: 46.4	F: 44.6 A: 47.3
Decision Tree	F: 51.3 A: 55.2	F: 46.4 A: 45.3	F: 47.3 A: 43.9
Average	F: 48.9 A: 55.7	F: 46.9 A: 47.9	F: 45.8 A: 47.9

Prompt-Based Topics

	Spelling Correction		
Model	Manual	Auto	None
Neural Net	F: 61.4 A: 56.1	F: 50.8 A: 57.1	F: 55.4 A: 50.0
Log. Reg.	F: 56.1 A: 68.4	F: 46.4 A: 62.5	F: 49.1 A: 52.7
Decision Tree	F: 53.5 A: 61.4	F: 50.0 A: 54.5	F: 46.5 A: 57.1
Average	F: 57.0 A: 62.0	F: 49.1 A: 58.0	F: 50.3 A: 53.3

Secondly, we note that the models using manual spelling correction tended to outperform models using automatic or no spelling correction, though this finding was less reliable. Since the "manual" spelling correction was done primarily using feedback from a word processor, it may be the case that had the essays been written digitally with spell check options available, many of the errors that were corrected would have been found by the student themselves. Future work will be necessary to determine if word processor spell check features are sufficient for this task.

4.! REFERENCES

[1] Graham, S. et al. 2015. Formative Assessment and Writing: A Meta-Analysis. *The Elementary School Journal*. 115, 4 (2015), 523–547.

[2] Matsumura, L.C. et al. 2002. Teacher Feedback, Writing Assignment Quality, and Third-Grade Students' Revision in Lower-And Higher-Achieving Urban Schools. *The Elementary School Journal*. 103, 1 (2002), 3.

[3] North Carolina School Report Cards: *http://www.ncpublicschools.org/*. Accessed: 2016-03-01.

[4] Purcell-Gates, V. et al. 2007. Learning to read and write genre-specific text: Roles of authentic experience and explicit teaching. *Reading Research Quarterly*. 42, 1 (2007), 8–45.

TutorSpace: Content-centric Platform for Enabling Blended Learning in Developing Countries

Kuldeep Yadav, Kundan Shrivastava, Ranjeet Kumar, Saurabh Srivastava, Om Deshmukh
Xerox Research Centre, India
kuldeep.r@xerox.com, om.deshmukh@xerox.com

ABSTRACT

One significant impact of the Massive Open Online Courses (MOOCs) phenomenon is that they have accelerated the widespread availability of quality education content. We refer to this content as the Open Educational Resources (OERs). It is our hypothesis that the OERs can be used to supplement classroom teaching for improved teacher efficiency and better student outcomes. We present a platform called TutorSpace which helps in curating OER content from multiple sources, integrating this content into a curricular setting in the context of what the lecturer is teaching and delivering it to students in a personalized way. A particular novelty of the TutorSpace platform is its capability for content-driven non-linear navigation of video content.

1. INTRODUCTION

The developing economies such as India, Brazil, China, etc face acute shortage of quality instructors, which is one of the primary reason for large number of unemployable graduates [2, 3]. Quality educational content (i.e. videos, slides, assignments) generated by the MOOCs can be potentially used to improve student learning and engagement in developing countries. However, instructors find it hard to use OER content directly in their course due to many reasons such as lack of context, no easy way of cross-source content aggregation, limited content search and curation capabilities of existing systems, and network bandwidth constraints. For example, $Alice$ is an instructor of an Algorithms course in XYZ university and she had taught some of the basic sorting algorithms to the students of her class. She wants to find specific videos for the "heap sort" algorithm concept, which can be given as an homework to the students. As, there would be different videos available online for this concept with varying duration, difficulty level, sources, etc. Alice is likely to spend a lot of time navigating through the available videos to finally select a video which suits her class' requirement.

We present a platform called *TutorSpace* that helps in searching and curating OER content from multiple sources, allows integration this content into a curricular setting in the context of what the lecturer is teaching and helps delivering it to students in a personalized way. TutorSpace uses advance multimedia concepts to support features such as quick and efficient video navigation, identification of topic transitions in a video, adding annotations on a video, etc. For the students, TutorSpace enables self-paced and ubiquitous learning where they can see course material posted by the instructor. TutorSpace also provides capabilities for students to share their notes, video bookmarks with their peers and discuss the topic of mutual interest in discussion forums.

2. TUTORSPACE PLATFORM

The proposed TutorSpace platform [1] provides content-centric capabilities to help instructors in the course curation. It allows instructors to have a digital presence of a classroom-based course, ability to search relevant course materials, and inclusion of selected education content in the curriculum. One of the key features of TutorSpace is that it provide a lecture planning workbench where the instructor can pool content from different sources and intersperse outside content with snippets of his/her pre-created content or classroom teaching. For students, TutorSpace enables self-paced and ubiquitous learning where they can see course material posted by the instructor. It also allows students to share their notes, video bookmarks with their peers and discuss topics of mutual interest in discussion forums. Some of the primary functional components of TutorSpace are as follows:

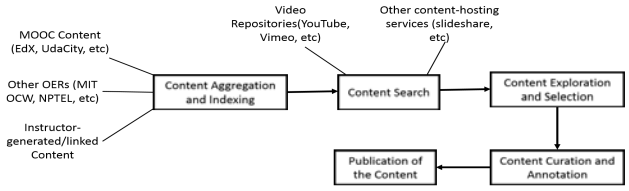
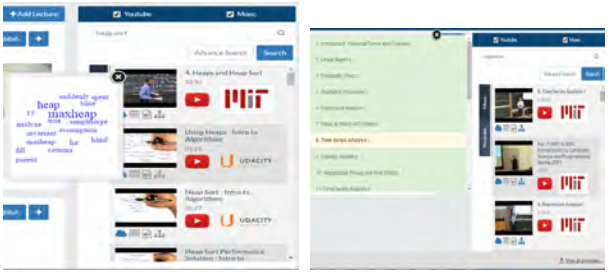


Figure 1: Step-by-step overview of instructor-led content curation and selection

2.1 Content Aggregation, Indexing & Search

TutorSpace aggregates content from different sources i.e. MOOCs (Coursera, EdX, Udacity), YouTube, etc. The content aggregation includes indexing meta-data about the course (i.e., information, syllabus), and video lecture specific meta-data (title, description, transcript of the video, duration, etc). Similarly, TutorSpace provides flexibility to the instructor to upload/link his/her own self-generated content too. Figure 2a presents a snapshot of the search dashboard in TutorSpace. Instructor can search for any concept and the system returns a set of relevant video lectures. The instructor has the flexibility to add search filters w.r.t. the source of the content (e.g., known-OER or all-YouTube) as well as other advanced filters such as duration, presentation style (e.g., slide or black-board), etc. Additionally, TutorSpace indexes meta-data about each video and further, this meta-data is presented to provide additional cues to the instructor as shown in Figure 2b. One of these cues is customized word-cloud which contain some of important concepts covered in the video (i.e., video preview). A detailed step-to-step creation process of customized word-cloud is presented in one of our earlier work [5]. These cues can help in the first-level decision making of whether to play a video or not. For example, word-cloud can help instructor in answering broad question about the video such as, "does this video contain algorithms for both linear and binary search" or "does this video explain heap sort with implementation



(a) (b)

Figure 2: (a) A snapshot of content search dashboard of TutorSpace. (b) Snapshot of concept relationship for a video

in Python programming language". In low bandwidth settings, it can save significant amount of time for the instructors [5].

2.2 Content Exploration and Selection

The instructors need to take a deep-dive and explore the content completely before including it in the teaching plan. Content exploration, specifically for a video, is a time-consuming task where often videos have long durations. The instructor can select any video for detailed exploration from the search results shown in Figure 2a. TutorSpace makes content exploration less time-consuming by providing techniques for non-linear navigation in a video with the help of customized word-cloud and parallel 2-D timeline as shown in Figure 3. Consider a video with the duration of nearly 60 minutes which discusses different sorting algorithms, the information provided by the customized word-cloud will include the name and time sequence of different algorithms along with other important terms discussed, which can help an instructor in getting a time-aware representation of a video [5]. Further, the customized word-cloud is interactive and instructor can click on any of keyword and its occurrences are highlighted on the 2-D timeline. The keyword occurrences represent different time instances where the keyword appears in the video. Further, mouse-hover event on any of these occurrences provide the context (i.e. an adjacent sentence) where a given keyword has been spoken. The click on any of occurrences will navigate the video to the point, where it was spoken in the video.

Figure 3: A snapshot of non-linear video navigation dashboard in TutorSpace with the help of customized word-cloud

Sometime, instructors may want to select a part of the content as opposed to the complete video. For example, in a 60 minute video on sorting algorithms, she may want to select only “merge sort" concept and share it with the students. TutorSpace enables partial selection of a content using its easy “video stripping" method. As shown in Figure 3, The instructor can move “start" and “end" (blue color) markers on the video timeline to highlight part of video content and click on “strip" button to select the content. After selecting the content, the instructor can drag and drop the content in their lecture plan as shown in Figure 4.

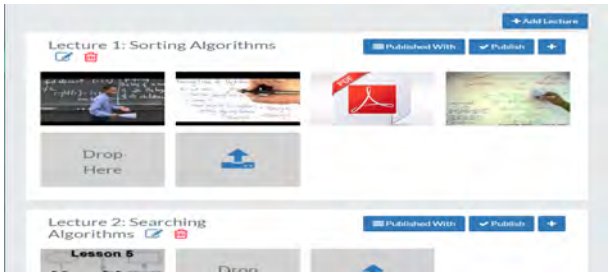


Figure 4: A snapshot of selected content (lecture plan) in TutorSpace

2.3 Other Features

TutorSpace provides a simple and user-friendly way to add notes and bookmarks on a video. After curation, the instructor can play the video and add annotations in terms of textual notes, images, external links/documents, etc with a click of a button. TutorSpace maintains detailed logs of interaction of the students with the content. It provides descriptive analytics on shared content to the instructors. The analytics include simple student-specific viewing statistics to fine-grained interaction pattern (i.e., time spent, pauses, play, etc). The instructor can use these findings to adapt the course curation strategies or to infer perceived difficulty of certain concepts. For example, if many students are spending a considerable amount of time on a specific portion of a video, it may need to be clarified during the class. Furthermore, TutorSpace provides standard learning management system (LMS) specific features such as course management, deadline creation and submission, quizzes, discussion forums, and student information management.

3. DISCUSSION

In developing countries such as India, quality of education is yet to improve substantially. We presented TutorSpace platform which can seamlessly enable integration of high-quality OER content in traditional classroom settings. TutorSpace provides rich multimedia capabilities w.r.t. content-indexing, search, non-linear navigation, and rich curation of the content. These capabilities are specifically designed to help instructor in developing countries. In our initial field-trial with the instructors, they appreciated the capabilities of the platform and provided several valuable feedback, which will be crucial for a long term acceptance of such a platform. We are in process of deploying TutorSpace to many engineering colleges in India and will be discussing our experiences in a future study.

4. REFERENCES

[1] TutorSpace project page, **http://xrci.xerox.com/tutorSpace-at-scale-personalized-learning**

[2] Cutrell, Edward et al. "Blended Learning in Indian Colleges with Massively Empowered Classroom." In Proceedings of the Second (2015) ACM Conference on Learning@ Scale, pp. 47-56. ACM, 2015.

[3] Chetlur, Malolan et al. "EduPaL: Enabling Blended Learning in Resource Constrained Environments." ACM DEV 2014.

[4] Guo, P. J., & Reinecke, K. (2014, March). Demographic differences in how students navigate through MOOCs. In Proceedings of the first ACM conference on Learning@ scale conference (pp. 21-30). ACM. Chicago

[5] Yadav, K. et al. Content-driven Multi-modal Techniques for Non-linear Video Navigation. In Proceedings of the 20th International Conference on Intelligent User Interfaces (pp. 333-344). ACM.